

Website@School Developer Documentation 1.0.0



Contents

Package wascore Procedural Elements	2
admin.php	2
config-example.php	3
cron.php	7
file.php	8
Define WASENTRY	8
index.php	9
init.php	10
Define MAXIMUM_ITERATIONS	10
Define THUMBNAIL_PREFIX	10
Define WLOG_ALERT	10
Define WLOG_CRIT	10
Define WLOG_DEBUG	11
Define WLOG_EMERG	11
Define WLOG_ERR	11
Define WLOG_INFO	11
Define WLOG_NOTICE	11
Define WLOG_WARNING	11
Function diff_microtime	11
Function error_exit	11
Function initialise	13
Function wasentry_script_name	13
Function was_version_check	13
admin.php	15
loginlib.php	16
was.php	17
admin.php	18
loginlib.php	19
was.php	20
accountmanagerlib.php	21
Define GROUPMANAGER_DIALOG_ADD	21
Define GROUPMANAGER_DIALOG_CAPACITY_ADMIN	21
Define GROUPMANAGER_DIALOG_CAPACITY_INTRANET	21
Define GROUPMANAGER_DIALOG_CAPACITY_PAGEMANAGER	21
Define GROUPMANAGER_DIALOG_DELETE	21
Define GROUPMANAGER_DIALOG_EDIT	21
Define TASK_ACCOUNTS	21
Define TASK_GROUPS	21
Define TASK_GROUP_ADD	21
Define TASK_GROUP_CAPACITY_ADMIN	21
Define TASK_GROUP_CAPACITY_INTRANET	21
Define TASK_GROUP_CAPACITY_MODULE	21
Define TASK_GROUP_CAPACITY_OVERVIEW	21

Define TASK_GROUP_CAPACITY_PAGEMANAGER	21
Define TASK_GROUP_CAPACITY_SAVE	21
Define TASK_GROUP_DELETE	21
Define TASK_GROUP_EDIT	21
Define TASK_GROUP_SAVE	21
Define TASK_GROUP_SAVE_NEW	21
Define TASK_USERS	21
Define TASK_USER_ADD	22
Define TASK_USER_ADMIN	22
Define TASK_USER_ADVANCED	22
Define TASK_USER_DELETE	22
Define TASK_USER_EDIT	22
Define TASK_USER_GROUPADD	22
Define TASK_USER_GROUPDELETE	22
Define TASK_USER_GROUPS	22
Define TASK_USER_GROUPSAVE	22
Define TASK_USER_INTRANET	22
Define TASK_USER_MODULE	22
Define TASK_USER_PAGEMANAGER	22
Define TASK_USER_SAVE	22
Define TASK_USER_SAVE_NEW	22
Define TASK_USER_TREEVIEW	22
Define USERMANAGER_DIALOG_ADD	22
Define USERMANAGER_DIALOG_ADMIN	22
Define USERMANAGER_DIALOG_DELETE	22
Define USERMANAGER_DIALOG_EDIT	22
Define USERMANAGER_DIALOG_INTRANET	22
Define USERMANAGER_DIALOG_PAGEMANAGER	22
Function job_accountmanager	22
Function show_accounts_intro	22
Function show_accounts_menu	22
aclmanager.class.php	24
Define ACL_LEVEL_AREA	24
Define ACL_LEVEL_NONE	24
Define ACL_LEVEL_PAGE	24
Define ACL_LEVEL_SECTION	24
Define ACL_LEVEL_SITE	24
Define ACL_TYPE_ADMIN	24
Define ACL_TYPE_INTRANET	24
Define ACL_TYPE_MODULE	25
Define ACL_TYPE_PAGEMANAGER	25
alertmanager.class.php	26
Define ALERTMANAGER_CHORE_ADD	26
Define ALERTMANAGER_CHORE_DELETE	26
Define ALERTMANAGER_CHORE_EDIT	26
Define ALERTMANAGER_CHORE_RULE_ADD	26
Define ALERTMANAGER_CHORE_RULE_DELETE	26
Define ALERTMANAGER_CHORE_RULE_EDIT	26
Define ALERTMANAGER_CHORE_RULE_SAVE	26

Define ALERTMANAGER CHORE SAVE	26
Define ALERTMANAGER CHORE VIEW	26
areamanager.class.php	27
Define AREAMANAGER CHORE ADD	27
Define AREAMANAGER CHORE DELETE	27
Define AREAMANAGER CHORE EDIT	27
Define AREAMANAGER CHORE EDIT THEME	27
Define AREAMANAGER CHORE RESET THEME	27
Define AREAMANAGER CHORE SAVE	27
Define AREAMANAGER CHORE SAVE NEW	27
Define AREAMANAGER CHORE SET DEFAULT	27
Define AREAMANAGER CHORE VIEW	27
Define AREAMANAGER DIALOG ADD	27
Define AREAMANAGER DIALOG DELETE	27
Define AREAMANAGER DIALOG EDIT	27
Define AREAMANAGER DIALOG EDIT THEME	27
Define AREAMANAGER DIALOG RESET THEME	27
configassistant.class.php	28
configurationmanagerlib.php	29
Define CHORE SAVE	29
Define TASK ALERTS	29
Define TASK AREAS	29
Define TASK CONFIGURATION INTRO	29
Define TASK SITE	29
Function job_configurationmanager	29
Function process task site	29
Function show_configuration_intro	30
Function show_configuration_menu	30
databaselib.php	31
Function database_factory	31
Function db_bool_is	32
Function db_delete	32
Function db_delete_sql	33
Function db_errormessage	33
Function db_escape_and_quote	33
Function db_insert_into	34
Function db_insert_into_and_get_id	34
Function db_insert_into_sql	35
Function db_last_insert_id	35
Function db_select_all_records	36
Function db_select_single_record	36
Function db_select_sql	37
Function db_update	38
Function db_update_sql	38
Function db_where_clause	39
mysql.class.php	40
mysqli.class.php	41
Define SQL_FALSE	41
Define SQL_TRUE	41

dbsessionlib.php	42
Function dbsession_close	43
Function dbsession_create	43
Function dbsession_destroy	44
Function dbsession_exists	44
Function dbsession_expire	45
Function dbsession_garbage_collection	45
Function dbsession_get_session_id	45
Function dbsession_open	46
Function dbsession_read	46
Function dbsession_remove_obsolete_sessions	46
Function dbsession_setup	47
Function dbsession_write	48
dialoglib.php	49
Define ATTR_CLASS_ERROR	52
Define ATTR_CLASS_VIEWONLY	52
Define BUTTON_CANCEL	52
Define BUTTON_DELETE	52
Define BUTTON_DONE	52
Define BUTTON_EDIT	52
Define BUTTON_GO	52
Define BUTTON_NO	52
Define BUTTON_OK	52
Define BUTTON_SAVE	52
Define BUTTON_YES	52
Define F_ALPHANUMERIC	52
Define F_CHECKBOX	53
Define F_CSRFTOKEN	53
Define F_DATE	53
Define F_DATETIME	53
Define F_FILE	53
Define F_INTEGER	53
Define F_LISTBOX	53
Define F_PASSWORD	53
Define F_RADIO	53
Define F_REAL	53
Define F_RICHTEXT	53
Define F_SUBMIT	53
Define F_TIME	53
Function accesskey_from_string	53
Function accesskey_tilde_to_underline	53
Function dialog_buttondef	54
Function dialog_csrf_token	54
Function dialog_get_class	54
Function dialog_get_label	55
Function dialog_get_widget	55
Function dialog_get_widget_file	56
Function dialog_get_widget_listbox	57
Function dialog_get_widget_radiocheckbox	58

Function dialog_get_widget_richtextinput	59
Function dialog_get_widget_submit	60
Function dialog_get_widget_textinput	61
Function dialog_quickform	62
Function dialog_validate	62
Function valid_datetime	63
donors.php	65
Function show_benefactor_logo	65
email.class.php	66
filelib.php	67
Function file_available	67
Function get_mediatype	67
Function get_mimetype	68
Function mime_ext_by_typ	69
Function mime_typ_by_ext	69
Function mime_typ_ext_match	70
filemanager.class.php	71
Define PARAM_FILENAME	71
Define PARAM_FILENAMES	71
Define PARAM_PATH	71
Define PARAM_SORT	71
Define SORTBY_DATE_ASC	71
Define SORTBY_DATE_DESC	71
Define SORTBY_FILE_ASC	71
Define SORTBY_FILE_DESC	71
Define SORTBY_NONE	71
Define SORTBY_SIZE_ASC	71
Define SORTBY_SIZE_DESC	71
Define TASK_ADD_DIRECTORY	71
Define TASK_ADD_FILE	71
Define TASK_CHANGE_DIRECTORY	71
Define TASK_LIST_DIRECTORY	71
Define TASK_PREVIEW_FILE	71
Define TASK_REMOVE_DIRECTORY	71
Define TASK_REMOVE_FILE	71
Define TASK_REMOVE_MULTIPLE_FILES	71
groupmanager.class.php	72
Define GROUPMANAGER_MAX_CAPACITIES	72
htmllib.php	73
Function href	73
Function html_a	73
Function html_attributes	74
Function html_form	75
Function html_form_close	75
Function html_img	75
Function html_input_radio	76
Function html_input_select	76
Function html_input_submit	76
Function html_input_text	76

Function html table	77
Function html table cell	77
Function html table cell close	77
Function html table close	77
Function html table head	77
Function html table head close	77
Function html table row	77
Function html table row close	78
Function html tag	78
Function html tag close	78
language.class.php	79
loginlib.php	80
Define BY EMAIL	82
Define BY LAISSEZ PASSER	82
Define BY PASSWORD	82
Define LOGIN FAILURE DELAY SECONDS	82
Define LOGIN PROCEDURE BLACKLIST	82
Define LOGIN PROCEDURE CHANGE PASSWORD	82
Define LOGIN PROCEDURE MESSAGE BOX	82
Define LOGIN PROCEDURE NORMAL	82
Define LOGIN PROCEDURE SEND BYPASS	82
Define LOGIN PROCEDURE SEND LAISSEZ PASSER	82
Define LOGIN PROCEDURE SHOWLOGIN	82
Define MINIMUM PASSWORD DIGITS	82
Define MINIMUM PASSWORD LENGTH	83
Define MINIMUM PASSWORD LOWERCASE	83
Define MINIMUM PASSWORD UPPERCASE	83
Function acceptable new password	83
Function authenticate user	84
Function login change password	85
Function login dialogdef	86
Function login failure blacklist address	86
Function login failure delay	87
Function login failure increment	87
Function login failure reset	87
Function login is blacklisted	88
Function login propagate navigation	88
Function login send bypass	89
Function login send confirmation	90
Function login send laissez passer	90
Function password hash check	91
Function password salt	92
Function show login	92
Function was login	94
Function was logout	95
Function was password hash	96
modulemanagerlib.php	97
Define TASK MODULEMANAGER EDIT	97
Define TASK MODULEMANAGER INTRO	97

Define TASK MODULEMANAGER SAVE	97
Function job_modulemanager	97
Function modulemanager cmp	97
Function modulemanager get_modules	97
Function modulemanager process	98
Function modulemanager show_intro	98
Function modulemanager show_menu	99
pagemanager.class.php	100
Define DIALOG_NODE_ADD	100
Define DIALOG_NODE_DELETE_CONFIRM	100
Define DIALOG_NODE_EDIT	100
Define DIALOG_NODE_EDIT_ADVANCED	100
Define DIALOG_NODE_EDIT_CONTENT	100
Define MODULE_NAME_DEFAULT	100
Define NODE_VISIBILIY_DEFAULT	100
Define NODE_VISIBILIY_EMBARGO	100
Define NODE_VISIBILIY_HIDDEN	100
Define NODE_VISIBILIY_VISIBLE	100
Define PARAM_SUBMENU_OPTION	100
Define PARAM_TREEVIEW	100
Define TASK_ADD_PAGE	100
Define TASK_ADD_SECTION	100
Define TASK_NODE_DELETE	100
Define TASK_NODE_EDIT	100
Define TASK_NODE_EDIT_ADVANCED	101
Define TASK_NODE_EDIT_CONTENT	101
Define TASK_NODE_FORCE_UNLOCK	101
Define TASK_PAGE_PREVIEW	101
Define TASK_SAVE_CONTENT	101
Define TASK_SAVE_NEWPAGE	101
Define TASK_SAVE_NEWSECTION	101
Define TASK_SAVE_NODE	101
Define TASK_SET_DEFAULT	101
Define TASK_SUBTREE_COLLAPSE	101
Define TASK_SUBTREE_EXPAND	101
Define TASK_TREEVIEW	101
Define TASK_TREEVIEW_SET	101
Define TREE_VIEW_CUSTOM	101
Define TREE_VIEW_MAXIMAL	101
Define TREE_VIEW_MINIMAL	101
statisticslib.php	102
Function job_statistics	102
Function statistics_embargo	102
Function statistics_get_pages	103
Function statistics_show_area_menu	103
theme.class.php	105
themelib.php	106
Function theme_factory	106
tokenlib.php	108

Function token create	108
Function token destroy	110
Function token fetch	110
Function token garbage collect	110
Function token lookup	111
Function token store	111
toolslib.php	113
Define CHORE SESSION DELETE	113
Define CHORE SESSION VIEW	113
Define TASK BACKUPTOOL	113
Define TASK LOGVIEW	113
Define TASK SESSIONTOOL	113
Define TASK TOOLS INTRO	113
Define TASK TRANSLATETOOL	113
Function job tools	113
Function logview download	113
Function logview priority	114
Function logview prune	114
Function logview show	115
Function sessions delete	115
Function sessions show	115
Function sessions show footer	116
Function sessions show header	116
Function sessions show node locked	117
Function sessions show overview	117
Function sessions show session	117
Function show tools intro	117
Function show tools menu	118
Function task backuptool	118
Function task logview	119
Function task sessiontool	119
Function tools get hostname	119
translatetool.class.php	120
Define TRANSLATETOOL CHORE EDIT	120
Define TRANSLATETOOL CHORE LANGUAGE ADD	120
Define TRANSLATETOOL CHORE LANGUAGE EDIT	120
Define TRANSLATETOOL CHORE LANGUAGE SAVE	120
Define TRANSLATETOOL CHORE LANGUAGE SAVE NEW	120
Define TRANSLATETOOL CHORE OVERVIEW	120
Define TRANSLATETOOL CHORE SAVE	120
Define TRANSLATETOOL PARAM DOMAIN	120
Define TRANSLATETOOL PARAM LANGUAGE KEY	120
updatelib.php	121
Define TASK INSTALL LANGUAGE	121
Define TASK INSTALL MODULE	121
Define TASK INSTALL THEME	121
Define TASK UPDATE CORE	121
Define TASK UPDATE LANGUAGE	122
Define TASK UPDATE MODULE	122

Define TASK_UPDATE_OVERVIEW	122
Define TASK_UPDATE_THEME	122
Function get_manifests	122
Function install_language	122
Function install_module	123
Function install_theme	123
Function job_update	124
Function update_core	125
Function update_core 2010120800	125
Function update_core 2010122100	125
Function update_core 2011020100	125
Function update_core 2011051100	125
Function update_core 2011093000	126
Function update_core 2012041900	129
Function update_core 2013071100	130
Function update_core 2014111700	130
Function update_core 2016062900	130
Function update_core version	130
Function update_create_table	131
Function update_create_tables	131
Function update_language	131
Function update_module	132
Function update_remove_obsolete_files	132
Function update_show_overview	133
Function update_status_anchor	133
Function update_status_table_close	133
Function update_status_table_open	133
Function update_theme	134
useraccount.class.php	135
Define ACL_ROLE_GURU	135
Define ACL_ROLE_INTRANET_ACCESS	135
Define ACL_ROLE_NONE	135
Define ACL_ROLE_PAGEMANAGER_AREAMASTER	135
Define ACL_ROLE_PAGEMANAGER_CONTENTMASTER	135
Define ACL_ROLE_PAGEMANAGER_PAGEMASTER	135
Define ACL_ROLE_PAGEMANAGER_SECTIONMASTER	135
Define ACL_ROLE_PAGEMANAGER_SITEMASTER	135
Define JOB_PERMISSION_ACCOUNTMANAGER	135
Define JOB_PERMISSION_BACKUPTOOL	135
Define JOB_PERMISSION_CONFIGURATIONMANAGER	136
Define JOB_PERMISSION_FILEMANAGER	136
Define JOB_PERMISSION_GURU	136
Define JOB_PERMISSION_LOGVIEW	136
Define JOB_PERMISSION_MASK	136
Define JOB_PERMISSION_MODULEMANAGER	136
Define JOB_PERMISSION_NEXT_AVAILABLE_VALUE	136
Define JOB_PERMISSION_PAGEMANAGER	136
Define JOB_PERMISSION_SESSIONTOOL	136
Define JOB_PERMISSION_STARTCENTER	136

Define JOB_PERMISSION_STATISTICS	136
Define JOB_PERMISSION_TOOLS	137
Define JOB_PERMISSION_TRANSLATETOOL	137
Define JOB_PERMISSION_UPDATE	137
Define PERMISSION_AREA_ADD_PAGE	137
Define PERMISSION_AREA_ADD_SECTION	137
Define PERMISSION_AREA_DROP_PAGE	137
Define PERMISSION_AREA_DROP_SECTION	137
Define PERMISSION_AREA_EDIT_AREA	137
Define PERMISSION_NODE_ADD_CONTENT	137
Define PERMISSION_NODE_ADD_PAGE	137
Define PERMISSION_NODE_ADD_SECTION	137
Define PERMISSION_NODE_DROP_CONTENT	137
Define PERMISSION_NODE_DROP_PAGE	137
Define PERMISSION_NODE_DROP_SECTION	137
Define PERMISSION_NODE_EDIT_CONTENT	137
Define PERMISSION_NODE_EDIT_PAGE	137
Define PERMISSION_NODE_EDIT_SECTION	137
Define PERMISSION_SITE_ADD_AREA	137
Define PERMISSION_SITE_DROP_AREA	137
Define PERMISSION_SITE_EDIT_SITE	137
usermanager.class.php	138
Define GROUP_SELECT_ALL_USERS	138
Define GROUP_SELECT_NO_GROUP	138
utf8lib.php	139
Define USE_MBSTRING	140
Function utf8_strcasecmp	140
Function utf8_strlen	140
Function utf8_strtoascii	140
Function utf8_strtolower	141
Function utf8_strtoupper	141
Function utf8_substr	142
Function utf8_validate	142
waslib.php	144
Define CAPACITY_CHAIR	144
Define CAPACITY_CUSTOM1	144
Define CAPACITY_CUSTOM2	144
Define CAPACITY_CUSTOM3	144
Define CAPACITY_CUSTOM4	144
Define CAPACITY_CUSTOM5	144
Define CAPACITY_CUSTOM6	144
Define CAPACITY_CUSTOM7	144
Define CAPACITY_CUSTOM8	144
Define CAPACITY_CUSTOM9	144
Define CAPACITY_EDITOR	144
Define CAPACITY_MEMBER	144
Define CAPACITY_NEXT_AVAILABLE	144
Define CAPACITY_NONE	144
Define CAPACITY_PRINCIPAL	144

Define CAPACITY PROJECTLEAD	144
Define CAPACITY PUBLISHER	144
Define CAPACITY PUPIL	144
Define CAPACITY SECRETARY	144
Define CAPACITY TEACHER	144
Define CAPACITY TREASURER	144
Define QUASI RANDOM DIGITS	144
Define QUASI RANDOM DIGITS UPPER	144
Define QUASI RANDOM DIGITS UPPER LOWER	144
Define QUASI RANDOM HEXDIGITS	144
Function appropriate legal notices	144
Function calculate uri shortcuts	145
Function calc user related acs	146
Function capacity name	146
Function convert to type	146
Function cron send queued alerts	147
Function friendly bookmark	148
Function get area records	148
Function get cookie string	149
Function get csrftoken	149
Function get editor names	149
Function get friendly parameter	150
Function get module records	150
Function get page address url	151
Function get parameter int	151
Function get parameter string	152
Function get properties	152
Function get requested area	153
Function get requested filename	153
Function get requested node	153
Function get skin names	154
Function get unique number	154
Function get user groups	154
Function hmac	155
Function ini_get_int	155
Function is_expired	155
Function is_under_embargo	156
Function javascript alert	157
Function lock record	157
Function lock record node	159
Function lock release	159
Function lock release node	160
Function logger	160
Function magic unquote	161
Function performance_get_queries	161
Function performance_get_seconds	162
Function quasi_random_string	162
Function queue_area_node_alert	162
Function quoted_printable	164

Function redirect and exit	165
Function replace_crlf	166
Function sanitise_filename	166
Function short_datim	167
Function string2time	167
Function t	167
Function tree_build	168
Function tree_visibility	169
Function userdir_delete	170
Function userdir_is_empty	170
Function was_file_url	170
Function was_node_url	171
Function was_url	172
zip.class.php	174
Define ZIP_TYPE_BUFFER	174
Define ZIP_TYPE_FILE	174
Define ZIP_TYPE_NONE	174
Define ZIP_TYPE_STREAM	174
main_admin.php	175
Define JOB_ACCOUNTMANAGER	175
Define JOB_CONFIGURATIONMANAGER	175
Define JOB_FILEBROWSER	175
Define JOB_FILEMANAGER	175
Define JOB_FLASHBROWSER	175
Define JOB_IMAGEBROWSER	175
Define JOB_MODULEMANAGER	176
Define JOB_PAGEMANAGER	176
Define JOB_STARTCENTER	176
Define JOB_STATISTICS	176
Define JOB_TOOLS	176
Define JOB_UPDATE	176
Function add_javascript_popup_function	176
Function add_javascript_select_url_function	176
Function admin_continue_session	177
Function admin_login	177
Function admin_logout_and_exit	178
Function admin_show_login_and_exit	178
Function get_current_skin	178
Function get_versioncheck_url	179
Function job_start	179
Function main_admin	179
Function non_admin_redirect_and_exit	180
main_cron.php	181
Function main_cron	182
main_file.php	183
Function download_source	183
Function download_source_tree	184
Function error_exit404	184
Function main_file	184

Function readfile_chunked	186
Function rfc1123date	186
Function send_file_from_datadir	186
main_index.php	189
Function calculate_area	189
Function calculate_default_page	190
Function calculate_node_id	190
Function main_index	191
Function module_load_view	191
Function module_view	192
Function update_statistics	192
Function update_view_count	193
manual.php	194
Function get_available_languages	194
Function get_available_manuals	194
Function show_manual	195
Function show_screen_choose_language	195
Function show_screen_download	196
utf8lib.php	197
zip.class.php	199
version.php	200
Define WAS_ORIGINAL	201
Define WAS_RELEASE	201
Define WAS_RELEASE_DATE	201
Define WAS_VERSION	201

[Package wascore Classes](#) 202

Class AclManager	202
Var \$acl_id	206
Var \$acl_type	207
Var \$area_view_areas_open	207
Var \$area_view_a_params	207
Var \$area_view_enabled	207
Var \$a_params_save	207
Var \$dialog	208
Var \$dialogdef	208
Var \$dialogdef_areas	208
Var \$dialogdef_areas_total	208
Var \$header	208
Var \$intro	209
Var \$output	209
Var \$pagination_a_params	209
Var \$pagination_enabled	209
Var \$pagination_limit	209
Var \$pagination_offset	210
Var \$pagination_total	210
Var \$related_acls	210
Constructor AclManager	210
Method calc_areas_total	211
Method dialog_tableform	211

Method enable_area_view	212
Method enable_pagination	213
Method get_dialogdef_admin	213
Method get_dialogdef_intranet	214
Method get_dialogdef_pagemanager	214
Method get_icon_area	215
Method get_permissions	215
Method get_permissions_areas	216
Method get_permissions_nodes_in_area	216
Method get_roles_intranet	217
Method get_roles_pagemanager	217
Method save_data	217
Method save_data_admin	217
Method save_data_intranet	217
Method save_data_pagemanager	217
Method save_data_permissions	218
Method set_action	218
Method set_dialog	218
Method set_header	218
Method set_intro	219
Method set_related_acs	219
Method show_dialog	219
Method show_dialog_admin	219
Method show_dialog_intranet	220
Method show_dialog_pagemanager	220
Method show_tree_walk	220
Method tree_build	221
Class AdminOutput	222
Var \$breadcrumbs	223
Var \$content	223
Var \$dtd	223
Var \$funnel_mode	223
Var \$helptopic	223
Var \$html_head	224
Var \$http_headers	224
Var \$menu	224
Var \$messages_bottom	224
Var \$messages_inline	224
Var \$messages_top	225
Var \$pagination	225
Var \$skin	225
Var \$subtitle	225
Var \$suppress_output	225
Var \$text_only	226
Var \$title	226
Constructor AdminOutput	226
Method add_breadcrumb	227
Method add_content	227
Method add_html_header	227

Method add http header	227
Method add menu	228
Method add message	228
Method add meta	228
Method add meta http equiv	228
Method add pagination	228
Method add pagination item	229
Method add popup bottom	230
Method add popup top	230
Method add stylesheet	230
Method get address	230
Method get bottomline	230
Method get breadcrumbs	231
Method get content	231
Method get div messages	231
Method get html	232
Method get html head	232
Method get lines	232
Method get lmth	233
Method get logo	233
Method get menu	233
Method get navigation	233
Method get pagination	234
Method get popups	235
Method get quickbottom	235
Method get quicktop	235
Method send headers	236
Method send output	236
Method set funnel mode	236
Method set helptopic	236
Method set suppress output	236
Class AdminSkin	237
Var \$icon height	237
Var \$icon path	237
Var \$icon width	238
Var \$knob height	238
Var \$knob width	238
Var \$name	238
Var \$stylesheets	238
Var \$text icons	239
Var \$text only	239
Constructor AdminSkin	239
Method get icon	239
Method get knob	240
Method get stylesheets	240
Method is text only	240
Class AlertManager	240
Var \$intervals	241
Var \$output	241

Var \$show_parent_menu	241
Constructor AlertManager	242
Method alerts_overview	242
Method alert_delete	242
Method alert_edit	242
Method alert_rule_delete	243
Method alert_rule_edit	243
Method alert_rule_save	243
Method alert_save	244
Method a_param	244
Method dialog_validate_alert	244
Method dialog_validate_rule	245
Method get_dialogdef_alert	245
Method get_dialogdef_rule	245
Method run	245
Method show_alert	245
Method show_parent_menu	246
Method show_rule	246
Method tree_walk	246
Class AreaManager	247
Var \$areas	248
Var \$output	248
Var \$show_parent_menu	248
Constructor AreaManager	248
Method area_add	248
Method area_delete	249
Method area_edit	249
Method area_edittheme	250
Method area_overview	250
Method area_resettheme	251
Method area_save	252
Method area_savenew	252
Method area_setdefault	252
Method a_param	253
Method count_existing_theme_properties	253
Method get_dialogdef_add_area	253
Method get_dialogdef_delete_area	254
Method get_dialogdef_edit_area	254
Method get_dialog_data	254
Method get_icon_delete	254
Method get_icon_edit	255
Method get_icon_home	255
Method get_options_themes	256
Method get_theme_records	256
Method reset_theme_defaults	256
Method show_edit_menu	257
Method show_parent_menu	257
Method sort_order_new_area	257
Class ConfigAssistant	258

Var \$dialogdef	262
Var \$dialogdef_hidden	262
Var \$fields	262
Var \$keyfield	263
Var \$language_domain	263
Var \$prefix	263
Var \$records	263
Var \$table	263
Var \$where	264
Constructor ConfigAssistant	264
Method get_dialogdef	264
Method get_extra	264
Method get_options_from_extra	265
Method save_data	265
Method show_dialog	265
Class DatabaseMysql	266
Var \$charset	266
Var \$collation	266
Var \$db_link	267
Var \$db_name	267
Var \$db_password	267
Var \$db_server	267
Var \$db_type	267
Var \$db_username	268
Var \$db_version	268
Var \$debug	268
Var \$engine	268
Var \$errno	268
Var \$error	269
Var \$prefix	269
Var \$query_counter	269
Var \$utf8_support	269
Constructor DatabaseMysql	269
Method check_engine	270
Method close	271
Method column_definition	271
Method concat	272
Method connect	274
Method create_table	274
Method create_table_sql	275
Method drop_table	275
Method dump	275
Method escape	276
Method exec	277
Method last_insert_id	277
Method mysql_utf8mb3	277
Method mysql_utf8_support	278
Method query	278
Method table_exists	279

<u>Class DatabaseMysqli</u>	279
<u>Var \$charset</u>	280
<u>Var \$collation</u>	280
<u>Var \$db_link</u>	280
<u>Var \$db_name</u>	280
<u>Var \$db_password</u>	280
<u>Var \$db_server</u>	281
<u>Var \$db_type</u>	281
<u>Var \$db_username</u>	281
<u>Var \$db_version</u>	281
<u>Var \$debug</u>	281
<u>Var \$engine</u>	282
<u>Var \$errno</u>	282
<u>Var \$error</u>	282
<u>Var \$prefix</u>	282
<u>Var \$query_counter</u>	282
<u>Var \$utf8_support</u>	283
<u>Constructor DatabaseMysqli</u>	283
<u>Method check_engine</u>	283
<u>Method close</u>	284
<u>Method column_definition</u>	285
<u>Method concat</u>	286
<u>Method connect</u>	287
<u>Method create_table</u>	288
<u>Method create_table_sql</u>	288
<u>Method drop_table</u>	289
<u>Method dump</u>	289
<u>Method escape</u>	290
<u>Method exec</u>	290
<u>Method last_insert_id</u>	291
<u>Method mysqli_utf8mb3</u>	291
<u>Method mysqli_utf8_support</u>	291
<u>Method query</u>	292
<u>Method table_exists</u>	293
<u>Class DatabaseMysqliResult</u>	293
<u>Var \$db_link</u>	293
<u>Var \$errno</u>	293
<u>Var \$error</u>	294
<u>Var \$num_rows</u>	294
<u>Var \$result</u>	294
<u>Constructor DatabaseMysqliResult</u>	294
<u>Method close</u>	294
<u>Method fetch_all</u>	294
<u>Method fetch_all_assoc</u>	295
<u>Method fetch_row</u>	295
<u>Method fetch_row_assoc</u>	295
<u>Class DatabaseMysqliResult</u>	295
<u>Var \$errno</u>	295
<u>Var \$error</u>	296

Var \$num_rows	296
Var \$result	296
Constructor DatabaseMysqlResult	296
Method close	296
Method fetch_all	297
Method fetch_all_assoc	297
Method fetch_row	297
Method fetch_row_assoc	297
Class Email	297
Var \$attachments	298
Var \$charset	298
Var \$eol	298
Var \$headers	298
Var \$mailcc	299
Var \$mailfrom	299
Var \$mailreplyto	299
Var \$mailto	299
Var \$max_length	299
Var \$messages	300
Var \$minimal	300
Var \$related	300
Var \$subject	300
Constructor Email	300
Method add_attachment	300
Method add_mailcc	301
Method add_message	301
Method add_related	302
Method boundary	303
Method encode_message	303
Method encode_part	304
Method encode_part_headers	304
Method is_7bit	305
Method prepare_body	305
Method reset_all	306
Method rfc2047_qchar	306
Method rfc2047_qstring	307
Method rfc5322_address	309
Method rfc5322_message_id	310
Method send	311
Method send_body	311
Method set_header	312
Method set_mailfrom	312
Method set_mailreplyto	313
Method set_mailto	313
Method set_message	313
Method set_subject	314
Class FileManager	314
Var \$areas	315
Var \$current_directory	315

Var \$ext allow browse	315
Var \$ext allow upload	315
Var \$job	316
Var \$output	316
Var \$show thumbnails	316
Var \$sort	316
Var \$usergroups	316
Var \$vpaths	317
Constructor FileManager	317
Method allowed_extensions	317
Method cmp_entries bydate asc	318
Method cmp_entries bydate desc	318
Method cmp_entries byfile asc	318
Method cmp_entries byfile desc	319
Method cmp_entries bysize asc	319
Method cmp_entries bysize desc	319
Method cmp_groups	320
Method delete_directory	320
Method delete_files	321
Method explode_path	321
Method file_url	321
Method get_dialogdef add_directory	322
Method get_dialogdef add_files	322
Method get_dialogdef confirm delete_directory	322
Method get_dialogdef confirm delete_files	323
Method get_entries	323
Method get_entries_areas	323
Method get_entries_groups	324
Method get_entries_root	324
Method get_entries_users	324
Method get_max_file_uploads	325
Method has_allowed_extension	325
Method human_readable_size	325
Method sanitise_filetype	325
Method save_uploaded_file	326
Method show_breadcrumbs	328
Method show_dialog_add_files	328
Method show_dialog_confirm delete_directory	328
Method show_dialog_confirm delete_files	329
Method show_directories	329
Method show_directories_and_files	330
Method show_file_as_thumbnail	331
Method show_list	332
Method show_menu	332
Method sort_entries	332
Method task_add_directory	333
Method task_add_file	333
Method task_change_directory	333
Method task_list_directory	333

Method task_preview_file	334
Method task_remove_directory	334
Method task_remove_file	334
Method task_remove_multiple_files	334
Method unique_filename	335
Method valid_path	335
Method virusscan	336
Method vname	337
Method vpath	337
Class GroupManager	337
Var \$groups	338
Var \$group_capacity_records	338
Var \$output	338
Var \$show_parent_menu	338
Constructor GroupManager	339
Method add_group_capacity	339
Method areas_expand_collapse	339
Method a_params	339
Method calc_acl_id	340
Method capacity_admin	340
Method capacity_intranet	340
Method capacity_overview	341
Method capacity_pagemanager	341
Method capacity_save	341
Method delete_group_capacities_records	341
Method get_dialogdef_add_group	342
Method get_dialogdef_edit_group	342
Method get_groupname	342
Method get_group_capacity_names	342
Method get_group_capacity_records	343
Method get_group_record	343
Method get_icon_delete	343
Method get_icon_edit	344
Method get_options_capacities	344
Method groups_overview	344
Method group_add	345
Method group_delete	345
Method group_edit	346
Method group_save	346
Method group_savenew	346
Method has_job_permission	347
Method show_breadcrumbs_addgroup	347
Method show_breadcrumbs_group	347
Method show_breadcrumbs_groupcapacity	348
Method show_menu_group	348
Method show_menu_groupcapacity	348
Method show_parent_menu	349
Method valid_group_capacity	349
Class Language	349

Var \$default_domain	350
Var \$languages	350
Var \$phrases	350
Constructor Language	350
Method get_active_language_names	350
Method get_current_language	351
Method get_filenames_to_try	351
Method get_languages_to_try	352
Method get_phrase	353
Method get_phrases_from_database	354
Method get_phrases_from_file	354
Method reset_cache	355
Method retrieve_languages	355
Class PageManager	356
Var \$areas	356
Var \$area_id	356
Var \$output	356
Var \$tree	357
Constructor PageManager	357
Method build_cached_tree	357
Method calculate_new_sort_order	358
Method calculate_updated_sort_order	358
Method calc_home_id	360
Method delete_node	360
Method get_dialogdef_add_node	360
Method get_dialogdef_edit_advanced_node	361
Method get_dialogdef_edit_node	361
Method get_dialogdef_force_unlock	362
Method get_dialog_data_node	362
Method get_icon_delete	362
Method get_icon_edit	363
Method get_icon_home	363
Method get_icon_invisibility	364
Method get_icon_page_preview	364
Method get_icon_section	365
Method get_link_node_edit	365
Method get_node_id_and_ancestors	366
Method get_options_area	366
Method get_options_modules	367
Method get_options_parents	367
Method get_options_parents_walk	368
Method get_options_sort_order	368
Method lock_records_subtree	369
Method message_from_lockinfo	370
Method module_connect	370
Method module_disconnect	371
Method module_load_admin	371
Method module_save	372
Method module_show_edit	373

Method node full name	373
Method node has grandchildren	374
Method permission add any node	374
Method permission add node	374
Method permission delete node	375
Method permission edit node	375
Method permission edit node content	376
Method permission set default	376
Method save node	377
Method save node new area mass move	378
Method section is open	379
Method show area menu	379
Method show dialog delete node confirm	380
Method show dialog force unlock	380
Method show edit menu	381
Method show tree	381
Method show treeview buttons	382
Method show tree walk	383
Method task force unlock	384
Method task node add	384
Method task node delete	384
Method task node edit	385
Method task node edit content	386
Method task page preview	386
Method task save content	388
Method task save newnode	388
Method task save node	389
Method task set default	389
Method task subtree collapse	389
Method task subtree expand	390
Method task treeview	390
Method task treeview set	391
Class Theme	391
Var \$area_id	391
Var \$area_record	391
Var \$breadcrumb_addendum	391
Var \$breadcrumb_separator	392
Var \$config	392
Var \$content	392
Var \$domain	392
Var \$dtd	392
Var \$html_head	393
Var \$http_headers	393
Var \$jumps	393
Var \$messages_bottom	393
Var \$messages_inline	394
Var \$messages_top	394
Var \$node_id	394
Var \$node_record	394

Var \$preview_mode	394
Var \$quickbottom_separator	395
Var \$quicktop_separator	395
Var \$silent_mode	395
Var \$text_only	395
Var \$theme_id	395
Var \$theme_record	396
Var \$title	396
Var \$tree	396
Constructor Theme	396
Method add_content	397
Method add_html_header	397
Method add_http_header	397
Method add_message	397
Method add_meta	397
Method add_meta_http_equiv	398
Method add_popup_bottom	398
Method add_popup_top	398
Method add_stylesheet	398
Method calc_breadcrumb_trail	399
Method construct_tree	399
Method dump_subtree	399
Method get_address	400
Method get_bazaar_style_style	400
Method get_bottomline	400
Method get_content	401
Method get_div_breadcrumbs	401
Method get_div_messages	401
Method get_html	402
Method get_html_head	402
Method get_jumpmenu	402
Method get_lines	403
Method get_logo	403
Method get_menu	404
Method get_navigation	404
Method get_popups	405
Method get_properties	405
Method get_quickbottom	405
Method get_quicklinks	406
Method get_quicktop	406
Method node2anchor	407
Method queue_alert	408
Method send_headers	408
Method send_output	408
Method set_preview_mode	408
Method show_tree_walk	409
Class TranslateTool	410
Var \$domains	410
Var \$languages	410

Var \$output	410
Var \$show_parent_menu	411
Constructor TranslateTool	411
Method a_param	411
Method code_highlight	411
Method diff_to_text	412
Method get_dialogdef_language	413
Method get_dialogdef_language_domain	413
Method get_domains	414
Method get_icon_edit	414
Method get_options_languages	415
Method get_strings_system	415
Method guess_row_count	416
Method languages_overview	416
Method language_add	417
Method language_edit	417
Method language_save	417
Method language_savenew	418
Method put_strings_userfile	418
Method render_translation_dialog	419
Method show_domain_menu	419
Method show_parent_menu	420
Method submit_diff_to_project	420
Method translation_edit	420
Method translation_save	420
Class Useraccount	421
Var \$acls	423
Var \$acls_areas	423
Var \$acls_modules	423
Var \$acls_modules_areas	423
Var \$acls_modules_nodes	424
Var \$acls_nodes	424
Var \$acl_id	424
Var \$area_permissions_from_nodes	424
Var \$editor	424
Var \$email	425
Var \$full_name	425
Var \$is_logged_in	425
Var \$language_key	425
Var \$path	425
Var \$properties	426
Var \$related_acls	426
Var \$skin	426
Var \$username	426
Var \$user_id	426
Constructor Useraccount	427
Method fetch_acls_from_table	427
Method has_area_permissions	427
Method has_intranet_permissions	428

Method has job permissions	428
Method has module area permissions	428
Method has module node permissions	428
Method has module site permissions	429
Method has node permissions	429
Method has site permissions	430
Method is admin	430
Method is admin pagemanager	430
Method where acl id	431
Class UserManager	431
Var \$output	431
Var \$show parent menu	431
Var \$users	432
Constructor UserManager	432
Method areas expand collapse	432
Method a params	432
Method calc acl id	433
Method delete user records	433
Method get dialogdef add user	433
Method get dialogdef add usergroup	433
Method get dialogdef confirm delete	434
Method get dialogdef edit user	434
Method get fname	434
Method get icon delete	434
Method get icon edit	435
Method get icon groupdelete	435
Method get num user records	435
Method get options available groups capacities	436
Method get user names	436
Method get user record	437
Method get user records	437
Method has job permission	437
Method show breadcrumbs adduser	438
Method show breadcrumbs overview	438
Method show breadcrumbs user	438
Method show menu overview	439
Method show menu user	439
Method show parent menu	440
Method users overview	440
Method user add	440
Method user admin	441
Method user delete	441
Method user edit	442
Method user groupadd	442
Method user groupdelete	442
Method user groups	443
Method user groupsave	443
Method user intranet	443
Method user pagemanager	443

Method user_save	444
Method user_savenew	444
Method user_save_basic	444
Class Zip	445
Var \$central_directory	447
Var \$Error	447
Var \$no_name_files	447
Var \$offset	447
Var \$zip_buffer	448
Var \$zip_comment	448
Var \$zip_filehandle	448
Var \$zip_path	448
Var \$zip_type	448
Constructor Zip	449
Method AddData	449
Method AddFile	449
Method CloseZip	449
Method dos_time_date	449
Method make_suitable_filename	450
Method OpenZipbuffer	450
Method OpenZipfile	451
Method OpenZipstream	451
Method zip_add_data	451
Method zip_add_directories	452
Method zip_error	452

[Package wasinstall Procedural Elements](#) 455

install.php	455
Define INSTALL_DIALOG_CANCELLED	455
Define INSTALL_DIALOG_CMS	455
Define INSTALL_DIALOG_COMPATIBILITY	455
Define INSTALL_DIALOG_CONFIRM	455
Define INSTALL_DIALOG_DATABASE	455
Define INSTALL_DIALOG_DONE	455
Define INSTALL_DIALOG_DOWNLOAD	455
Define INSTALL_DIALOG_FINISH	455
Define INSTALL_DIALOG_INSTALLTYPE	455
Define INSTALL_DIALOG_LANGUAGE	455
Define INSTALL_DIALOG_LICENSE	455
Define INSTALL_DIALOG_USER	456
Define PROJECT_SITE	456
Define WASENTRY	456
Function install_script_name	456
demodata.php	458
Function demodata	458
Function demodata_alerts	459
Function demodata_areas	459
Function demodata_sections_pages	460
Function demodata_users_groups	460
index.php	462

demodata.php	463
install.php	464
demodata.php	465
install.php	466
tabledata.php	467
tabledefs.php	468
Package wasinstall Classes	470
Class InstallWizard	470
Var \$license	471
Var \$messages	471
Var \$results	471
Var \$time_start	471
Constructor InstallWizard	472
Method appropriate_legal_notices	472
Method button	473
Method check_compatibility	473
Method check_for_nameclash	473
Method check_license	473
Method check_validation	474
Method clamscan_installed	474
Method construct_config_php	474
Method create_tables	474
Method diff_microtime	475
Method end_session_and_redirect	475
Method errorcount_bump	475
Method errorcount_reset	475
Method fetch_license	475
Method gd_supported	476
Method get_default_install_values	477
Method get_dialogdef_cms	477
Method get_dialogdef_database	477
Method get_dialogdef_finish	477
Method get_dialogdef_installtype	478
Method get_dialogdef_language	478
Method get_dialogdef_user	478
Method get_list_of_install_languages	478
Method get_manifests	478
Method get_menu	479
Method get_options_db_type	480
Method get_page	480
Method get_utf8_parameter_string	481
Method guess_url	482
Method insert_tabledata	482
Method invisible_test_image	482
Method is_already_installed	483
Method magic_unquote	484
Method microtime	484
Method mysql_close	484
Method mysql_connect	484

Method mysql_get_server_info	484
Method mysql_num_rows	485
Method mysql_query	485
Method mysql_real_escape_string	485
Method mysql_select_db	486
Method mysql_set_charset	486
Method mysql_utf8mb3	486
Method mysql_utf8_support	486
Method perform_installation	487
Method quasi_random_string	488
Method render_dialog	488
Method run	489
Method sanitise_filename	489
Method save_cms	490
Method save_database	490
Method save_installtype	491
Method save_language	491
Method save_user	491
Method show_dialog_cancelled	491
Method show_dialog_cms	492
Method show_dialog_compatibility	492
Method show_dialog_confirm	493
Method show_dialog_database	493
Method show_dialog_finish	494
Method show_dialog_installtype	494
Method show_dialog_language	494
Method show_dialog_license	495
Method show_dialog_user	495
Method t	496
Method validate	496
Method validate_password	497
Method write_config_php	498
Package waslang_ar Procedural Elements	500
demodata.php	500
install.php	501
admin.php	502
ar_manifest.php	503
loginlib.php	504
was.php	505
htmlpage.php	506
sitemap.php	507
axis.php	508
frugal.php	509
rosalina.php	510
schoolyard.php	511
Package waslang_bg Procedural Elements	513
demodata.php	513
install.php	514

admin.php	515
bg_manifest.php	516
loginlib.php	517
was.php	518
aggregator.php	519
althing.php	520
crew.php	521
htmlpage.php	522
mailpage.php	523
redirect.php	524
sitemap.php	525
snapshots.php	526
axis.php	527
cornelia.php	528
frugal.php	529
rosalina.php	530
schoolyard.php	531
sophia.php	532
Package waslang_da Procedural Elements	534
demodata.php	534
install.php	535
admin.php	536
da_manifest.php	537
loginlib.php	538
was.php	539
htmlpage.php	540
sitemap.php	541
axis.php	542
frugal.php	543
rosalina.php	544
schoolyard.php	545
Package waslang_de Procedural Elements	547
demodata.php	547
install.php	548
admin.php	549
de_manifest.php	550
loginlib.php	551
was.php	552
aggregator.php	553
crew.php	554
htmlpage.php	555
mailpage.php	556
redirect.php	557
sitemap.php	558
snapshots.php	559
axis.php	560
cornelia.php	561
frugal.php	562

rosalina.php	563
schoolyard.php	564
sophia.php	565
Package waslang_el Procedural Elements	567
demodata.php	567
install.php	568
admin.php	569
el_manifest.php	570
loginlib.php	571
was.php	572
aggregator.php	573
althing.php	574
confab.php	575
crew.php	576
htmlpage.php	577
redirect.php	578
sitemap.php	579
axis.php	580
frugal.php	581
rosalina.php	582
schoolyard.php	583
Package waslang_en Procedural Elements	585
en_manifest.php	585
Package waslang_es Procedural Elements	587
demodata.php	587
install.php	588
admin.php	589
es_manifest.php	590
loginlib.php	591
was.php	592
aggregator.php	593
crew.php	594
htmlpage.php	595
sitemap.php	596
axis.php	597
cornelia.php	598
frugal.php	599
rosalina.php	600
schoolyard.php	601
sophia.php	602
Package waslang_fa Procedural Elements	604
demodata.php	604
install.php	605
admin.php	606
fa_manifest.php	607
loginlib.php	608
was.php	609
htmlpage.php	610

sitemap.php	611
axis.php	612
frugal.php	613
rosalina.php	614
schoolyard.php	615
Package waslang_fi Procedural Elements	617
demodata.php	617
install.php	618
admin.php	619
fi_manifest.php	620
loginlib.php	621
was.php	622
htmlpage.php	623
sitemap.php	624
axis.php	625
frugal.php	626
rosalina.php	627
schoolyard.php	628
Package waslang_fil Procedural Elements	630
admin.php	630
fil_manifest.php	631
loginlib.php	632
was.php	633
Package waslang_fr Procedural Elements	635
demodata.php	635
install.php	636
admin.php	637
fr_manifest.php	638
loginlib.php	639
was.php	640
aggregator.php	641
crew.php	642
htmlpage.php	643
mailpage.php	644
redirect.php	645
sitemap.php	646
snapshots.php	647
axis.php	648
cornelia.php	649
frugal.php	650
rosalina.php	651
schoolyard.php	652
sophia.php	653
Package waslang_fry Procedural Elements	655
demodata.php	655
install.php	656
admin.php	657
fry_manifest.php	658

loginlib.php	659
was.php	660
aggregator.php	661
althing.php	662
confab.php	663
crew.php	664
htmlpage.php	665
mailpage.php	666
redirect.php	667
sitemap.php	668
snapshots.php	669
axis.php	670
cornelia.php	671
frugal.php	672
rosalina.php	673
schoolyard.php	674
sophia.php	675
Package waslang_hi Procedural Elements	677
demodata.php	677
admin.php	678
hi_manifest.php	679
loginlib.php	680
was.php	681
htmlpage.php	682
sophia.php	683
Package waslang_hu Procedural Elements	685
demodata.php	685
install.php	686
admin.php	687
hu_manifest.php	688
loginlib.php	689
was.php	690
htmlpage.php	691
sitemap.php	692
axis.php	693
frugal.php	694
rosalina.php	695
schoolyard.php	696
Package waslang_it Procedural Elements	698
demodata.php	698
install.php	699
admin.php	700
it_manifest.php	701
loginlib.php	702
was.php	703
aggregator.php	704
crew.php	705
htmlpage.php	706

mailpage.php	707
redirect.php	708
sitemap.php	709
snapshots.php	710
axis.php	711
cornelia.php	712
frugal.php	713
rosalina.php	714
schoolyard.php	715
sophia.php	716
Package waslang_nl Procedural Elements	718
nl_manifest.php	718
Package waslang_pap Procedural Elements	720
demodata.php	720
install.php	721
admin.php	722
loginlib.php	723
pap_manifest.php	724
was.php	725
aggregator.php	726
crew.php	727
htmlpage.php	728
mailpage.php	729
redirect.php	730
sitemap.php	731
snapshots.php	732
axis.php	733
cornelia.php	734
frugal.php	735
rosalina.php	736
schoolyard.php	737
sophia.php	738
Package waslang_pl Procedural Elements	740
demodata.php	740
install.php	741
admin.php	742
loginlib.php	743
pl_manifest.php	744
was.php	745
htmlpage.php	746
sitemap.php	747
axis.php	748
frugal.php	749
rosalina.php	750
schoolyard.php	751
Package waslang_pt Procedural Elements	753
demodata.php	753
install.php	754

admin.php	755
loginlib.php	756
pt_manifest.php	757
was.php	758
aggregator.php	759
crew.php	760
htmlpage.php	761
mailpage.php	762
redirect.php	763
sitemap.php	764
snapshots.php	765
axis.php	766
cornelia.php	767
frugal.php	768
rosalina.php	769
schoolyard.php	770
sophia.php	771
Package waslang_ro Procedural Elements	773
admin.php	773
loginlib.php	774
ro_manifest.php	775
was.php	776
htmlpage.php	777
sitemap.php	778
axis.php	779
frugal.php	780
rosalina.php	781
Package waslang_ru Procedural Elements	783
demodata.php	783
install.php	784
admin.php	785
loginlib.php	786
ru_manifest.php	787
was.php	788
aggregator.php	789
crew.php	790
htmlpage.php	791
mailpage.php	792
redirect.php	793
sitemap.php	794
snapshots.php	795
axis.php	796
cornelia.php	797
frugal.php	798
rosalina.php	799
schoolyard.php	800
sophia.php	801
Package waslang_sh Procedural Elements	803

admin.php	803
loginlib.php	804
sh_manifest.php	805
was.php	806
Package waslang_sk Procedural Elements	808
admin.php	808
loginlib.php	809
sk_manifest.php	810
was.php	811
aggregator.php	812
htmlpage.php	813
Package default Procedural Elements	815
ckeditor.php	815
spellchecker.php	816
Function error_handler	816
Function escape_quote	816
Function print_checker_results	816
Function print_suggs_elem	816
Function print_textindex_decl	816
Function print_textinputs_var	816
Function print_words_elem	816
fckeditor.php	818
fckeditor_php4.php	819
Function FCKEditor_IsCompatibleBrowser	819
crewserver.php	820
Define CREW_SERVER_DATE	820
Define CREW_SERVER_NAME	820
Define CREW_SERVER_VERSION	820
Function agplv3_compliance	820
Function dump_buffer	820
Function get_org_property	820
Function hexdump	820
Function hmac	820
Function initialise	821
Function logger	821
Function main	821
Function read_config	821
Function sockerr	821
Function utf16_strlen	821
Package default Classes	822
Class CKEditor	822
Var \$basePath	822
Var \$config	822
Var \$initialized	823
Var \$returnOutput	823
Var \$textareaAttributes	823
Var \$timestamp	823
Var \$version	823

Var \$ events	824
Var \$ globalEvents	824
Var \$ timestamp	824
Constructor CKEditor	824
Method addEventHandler	824
Method addGlobalEventHandler	825
Method ckeditorPath	825
Method clearEventHandlers	825
Method clearGlobalEventHandlers	825
Method configSettings	826
Method editor	826
Method init	826
Method jsEncode	826
Method replace	827
Method replaceAll	827
Method returnGlobalEvents	827
Method script	827
Class CrewClient	828
Var \$attr	828
Var \$authenticated	828
Var \$buf_in	828
Var \$buf_out	828
Var \$cid	828
Var \$date	828
Var \$headers	828
Var \$local_address	828
Var \$local_port	828
Var \$name	828
Var \$nick	829
Var \$range	829
Var \$remote_address	829
Var \$remote_port	829
Var \$server	829
Var \$socket	829
Var \$state	829
Var \$wid	829
Var \$workshop	829
Constructor CrewClient	829
Method frame_available	829
Method frame_decode	829
Method process_request	830
Method recv	830
Method send	830
Method valid_authentication	830
Method valid_handshake	830
Class CrewServer	831
Var \$cids	831
Var \$clients	831
Var \$dirty_sockets	831

Var \$mark_next	831
Var \$mark_time	831
Var \$server_address	831
Var \$server_port	831
Var \$server_run_flag	831
Var \$server_socket	831
Var \$sockets	831
Var \$wids	832
Var \$workshops	832
Constructor CrewServer	832
Method disconnect_client	832
Method disconnect_socket	832
Method dump_overview	832
Method find_workshop	832
Method frame_encode	832
Method get_tv_sec	833
Method initialise	833
Method lookup_client_cid	833
Method run	833
Class CrewWorkshop	833
Var \$attr	833
Var \$clients	833
Var \$name	833
Var \$origin	833
Var \$text	833
Var \$wid	834
Constructor CrewWorkshop	834
Method calc_jranges	834
Method cast_message	834
Method cast_userlist	834
Method cast_user_enters	834
Method cast_user_leaves	834
Method disjoin	834
Method get_next_attr	834
Method get_userlist	834
Method join	835
Method process_diff	835
Method send	835
Method send_text_full	835
Method send_userlist	835
Method send_user_info	835
Class FCKeditor	836
Var \$BasePath	836
Var \$Config	836
Var \$Height	836
Var \$InstanceName	836
Var \$ToolbarSet	837
Var \$Value	837
Var \$Width	837

Constructor FCKeditor	837
Method Create	837
Method CreateHtml	837
Method EncodeConfig	837
Method GetConfigFieldString	838
Method IsCompatible	838
Package waslang_sv Procedural Elements	840
demodata.php	840
install.php	841
admin.php	842
loginlib.php	843
sv_manifest.php	844
was.php	845
aggregator.php	846
crew.php	847
htmlpage.php	848
mailpage.php	849
redirect.php	850
sitemap.php	851
snapshots.php	852
axis.php	853
cornelia.php	854
frugal.php	855
rosalina.php	856
schoolyard.php	857
sophia.php	858
Package waslang_tr Procedural Elements	860
install.php	860
admin.php	861
loginlib.php	862
tr_manifest.php	863
was.php	864
axis.php	865
cornelia.php	866
Package waslang_ur Procedural Elements	868
demodata.php	868
install.php	869
admin.php	870
loginlib.php	871
ur_manifest.php	872
was.php	873
aggregator.php	874
crew.php	875
htmlpage.php	876
mailpage.php	877
redirect.php	878
sitemap.php	879
snapshots.php	880

axis.php	881
cornelia.php	882
frugal.php	883
rosalina.php	884
schoolyard.php	885
sophia.php	886
Package waslang_vi Procedural Elements	888
demodata.php	888
install.php	889
admin.php	890
loginlib.php	891
vi_manifest.php	892
was.php	893
aggregator.php	894
althing.php	895
confab.php	896
crew.php	897
htmlpage.php	898
mailpage.php	899
redirect.php	900
sitemap.php	901
snapshots.php	902
axis.php	903
cornelia.php	904
frugal.php	905
rosalina.php	906
schoolyard.php	907
sophia.php	908
Package waslang_zh Procedural Elements	910
demodata.php	910
install.php	911
admin.php	912
loginlib.php	913
was.php	914
zh_manifest.php	915
htmlpage.php	916
sitemap.php	917
axis.php	918
frugal.php	919
rosalina.php	920
schoolyard.php	921
Package wasmod_aggregator Procedural Elements	923
aggregator_admin.php	923
Function aggregator_check_node_list	923
Function aggregator_connect	924
Function aggregator_disconnect	924
Function aggregator_get_dialogdef	925
Function aggregator_save	925

Function aggregator show edit	926
aggregator cron.php	927
Function aggregator cron	927
aggregator install.php	928
Function aggregator demodata	928
Function aggregator install	929
Function aggregator uninstall	929
Function aggregator upgrade	930
aggregator manifest.php	931
aggregator search.php	932
Function aggregator search	932
aggregator view.php	933
Function aggregator view	933
Function aggregator view get config	934
Function aggregator view get modules	935
Function aggregator view get nodes	935
Function aggregator view get node from db	935
Function aggregator view get node from tree	935
Function aggregator view htmlpage	936
Function aggregator view snapshots	936
aggregator tabledefs.php	938
aggregator.php	939
aggregator.php	940

Package wasmod althing Procedural Elements	942
althing admin.php	942
Function althing config dialog validate	942
Function althing connect	943
Function althing disconnect	943
Function althing get dialogdef configuration	943
Function althing get dialogdef moderation	944
Function althing get dialogdef report	944
Function althing get post records	944
Function althing moderation dialog validate	945
Function althing report dialog validate	945
Function althing save	945
Function althing save configuration	946
Function althing save moderation	946
Function althing save report	947
Function althing save viewlog	947
Function althing show content	948
Function althing show edit	948
Function althing show edit configuration	949
Function althing show edit moderation	949
Function althing show edit moderation list	951
Function althing show edit moderation post	951
Function althing show edit report	952
Function althing show edit viewlog	952
althing common.php	954
Define ALTHING_DEFAULT_VISIBILITY	954

Define ALTHING_DEFAULT_VISIBILITY_HIDDEN	954
Define ALTHING_DEFAULT_VISIBILITY_REGISTERED	954
Define ALTHING_DEFAULT_VISIBILITY_VISIBLE	954
Define ALTHING_HARVEST_LIMIT	954
Define ALTHING_MARBLES_LIMIT	954
Define ALTHING_STATUS_CLOSED	954
Define ALTHING_STATUS_FROZEN	954
Define ALTHING_STATUS_OPENED	954
Function althing_get_emails	954
Function althing_send_alerts	954
Function bbcode2html	955
Function bbcode_callback	956
Function bbcode_callback_q	957
Function bbcode_help	957
Function bbcode_validate	957
Function bbcode_valid_path	958
althing_cron.php	959
Function althing_cron	959
althing_install.php	960
Function althing_demodata	960
Function althing_install	961
Function althing_uninstall	961
Function althing_upgrade	962
althing_manifest.php	963
althing_search.php	964
Function althing_search	964
Function althing_search_results	965
althing_view.php	966
Define ALTHING_DIRNAME	966
Define ALTHING_REFERENCE	966
Function althing_show_form	966
Function althing_show_overview	967
Function althing_show_preview	967
Function althing_show_thankyou	967
Function althing_submit_message	968
Function althing_view	969
Function althing_view_dialog_validate	969
Function althing_view_get_dialogdef	969
Function althing_view_get_posts	970
Function althing_view_show_post	970
Function check_add_reply_harvest	971
althing_tabledefs.php	973
althing.php	974
althing.php	975
Package wasmod_confab_Procedural Elements	977
confab_admin.php	977
Function confab_config_dialog_validate	977
Function confab_connect	978
Function confab_disconnect	978

Function confab_get_conversations	978
Function confab_get_dialogdef_configuration	979
Function confab_get_dialogdef_moderation	979
Function confab_get_dialogdef_report	979
Function confab_save	979
Function confab_save_configuration	980
Function confab_save_moderation	981
Function confab_save_report	981
Function confab_show_content	981
Function confab_show_edit	982
Function confab_show_edit_configuration	982
Function confab_show_edit_moderation	983
Function confab_show_edit_moderation_list	984
Function confab_show_edit_moderation_messages	985
Function confab_show_edit_report	985
confab_common.php	987
Define CONFAB_ACTION_MESSAGE	987
Define CONFAB_ACTION_REMOVE	987
Define CONFAB_ANONYMOUS_POSTFIX	987
Define CONFAB_BRAILLE	987
Define CONFAB_MAX_CONVERSATION_LIMIT	987
Define CONFAB_MAX_PARTICIPANTS_LIMIT	987
Define CONFAB_MAX_TRIES_LIMIT	987
Define CONFAB_PERSONAL_NO	987
Define CONFAB_PERSONAL_REGISTERED	987
Define CONFAB_PERSONAL_YES	987
Define CONFAB_REFRESH_LIMIT	987
Define CONFAB_STATUS_CLOSED	987
Define CONFAB_STATUS_OPENED	987
Define CONFAB_STATUS_REGISTERED	987
Define CONFAB_USER_STATUS_JOINED	987
Define CONFAB_USER_STATUS_LEAVING	987
Define CONFAB_USER_STATUS_NONE	987
Define CONFAB_VISUAL	987
Define CONFAB_VISUAL_BW	987
Define CONFAB_VISUAL_BY	987
Define CONFAB_VISUAL_RB	987
Function confab_conversation_timeout	987
Function confab_get_participants	988
Function confab_remote_addr	988
confab_cron.php	989
Function confab_cron	989
confab_install.php	990
Function confab_demodata	990
Function confab_install	991
Function confab_uninstall	991
Function confab_upgrade	992
confab_manifest.php	993
confab_search.php	994

Function confab_search	994
confab_view.php	996
Define CONFAB_DIRNAME	996
Define CONFAB_REFERENCE	996
Function confab_braille_dispatcher	996
Function confab_braille_page_headers	997
Function confab_braille_show_all	997
Function confab_braille_show_main	998
Function confab_braille_show_talk	998
Function confab_braille_show_users	998
Function confab_join	999
Function confab_join_conversation	999
Function confab_join_dialog_show	1000
Function confab_join_dialog_validate	1000
Function confab_join_get_dialogdef	1001
Function confab_leave_conversation	1002
Function confab_save_message	1002
Function confab_view	1002
Function confab_view_calc_participants	1003
Function confab_view_delete_stale_participants	1003
Function confab_view_get_messages	1004
Function confab_view_participants_timeout	1004
Function confab_visual_dispatcher	1005
Function confab_visual_show_main	1005
Function confab_visual_update	1006
confab_tabledefs.php	1008
confab.php	1009
confab.php	1010

Package wasmod_crew_Procedural Elements	1012
crew_admin.php	1012
Define CREW_ACL_ROLE_READONLY	1012
Define CREW_ACL_ROLE_READWRITE	1012
Define CREW_PERMISSION_READ	1012
Define CREW_PERMISSION_WRITE	1012
Function crew_connect	1012
Function crew_disconnect	1013
Function crew_get_dialogdef	1013
Function crew_save	1014
Function crew_show_edit	1015
crew_cron.php	1017
Function crew_cron	1017
crew_install.php	1018
Function crew_demodata	1018
Function crew_install	1019
Function crew_uninstall	1019
Function crew_upgrade	1020
crew_manifest.php	1021
crew_search.php	1022
Function crew_search	1022

crew_view.php	1024
Define CREW_MAX_DOCUMENT_SIZE	1024
Function crew_screen	1024
Function crew_view	1025
Function crew_view_dialogdef	1025
Function crew_view_get_workshop_data	1026
Function crew_view_save_document	1026
Function crew_view_show_edit	1026
Function crew_view_show_view	1027
crew_tabledefs.php	1029
crew.php	1030
crew.php	1031
Package wasmod_guestbook Procedural Elements	1033
guestbook_admin.php	1033
Function guestbook_connect	1033
Function guestbook_disconnect	1034
Function guestbook_save	1034
Function guestbook_show_edit	1035
guestbook.php	1037
guestbook.php	1038
Package wasmod_htmlpage Procedural Elements	1040
htmlpage_admin.php	1040
Function htmlpage_connect	1040
Function htmlpage_disconnect	1041
Function htmlpage_get_dialogdef	1041
Function htmlpage_save	1041
Function htmlpage_show_edit	1042
htmlpage_cron.php	1044
Function htmlpage_cron	1044
htmlpage_install.php	1045
Function htmlpage_demodata	1045
Function htmlpage_install	1046
Function htmlpage_uninstall	1046
Function htmlpage_upgrade	1046
htmlpage_manifest.php	1048
htmlpage_search.php	1049
Function htmlpage_search	1049
htmlpage_view.php	1052
Function htmlpage_view	1052
htmlpage_tabledefs.php	1053
htmlpage.php	1054
htmlpage.php	1055
Package wasmod_mailpage Procedural Elements	1057
mailpage_tabledefs.php	1057
mailpage.php	1058
mailpage.php	1059
mailpage_admin.php	1060
Function mailpage_connect	1060

Function mailpage dialog validate address	1061
Function mailpage disconnect	1061
Function mailpage get addresses	1061
Function mailpage get dialogdef address	1062
Function mailpage get dialogdef config	1062
Function mailpage get dialogdef delete	1062
Function mailpage get icon delete	1063
Function mailpage save	1063
Function mailpage save configuration	1064
Function mailpage save destinations	1064
Function mailpage show content	1065
Function mailpage show edit	1065
Function mailpage show edit configuration	1066
Function mailpage show edit destinations	1066
Function mailpage show edit destinations overview	1068
mailpage cron.php	1069
Function mailpage cron	1069
mailpage install.php	1070
Function mailpage demodata	1070
Function mailpage install	1071
Function mailpage uninstall	1071
Function mailpage upgrade	1071
mailpage manifest.php	1073
mailpage search.php	1074
Function mailpage search	1074
mailpage view.php	1075
Define MAILPAGE REFERENCE	1075
Function mailpage send message	1075
Function mailpage show form	1076
Function mailpage show preview	1076
Function mailpage show thankyou	1077
Function mailpage view	1077
Function mailpage view dialog validate	1077
Function mailpage view get config	1078
Function mailpage view get dialogdef	1078

Package wasmod mypage Procedural Elements	1080
mypage.php	1080
mypage.php	1081
mypage admin.php	1082
Function mypage connect	1082
Function mypage disconnect	1082
Function mypage save	1083
Function mypage show edit	1084
mypage cron.php	1085
Function mypage cron	1085
mypage install.php	1086
Function mypage demodata	1086
Function mypage install	1087
Function mypage uninstall	1087

Function mypage_upgrade	1087
mypage_manifest.php	1089
mypage_search.php	1090
Function mypage_search	1090
mypage_view.php	1091
Function mypage_authorisation	1091
Function mypage_password_dialog_validate	1092
Function mypage_password_get_dialogdef	1092
Function mypage_profile_dialog_validate	1092
Function mypage_profile_get_dialogdef	1092
Function mypage_profile_save	1093
Function mypage_view	1093
Function mypage_view_home	1094
Function mypage_view_login	1094
Function mypage_view_password	1094
Function mypage_view_profile	1095

[Package wasmod_newsletter Procedural Elements](#) 1098

newsletter_tabledefs.php	1098
newsletter.php	1099
newsletter.php	1100
newsletter_admin.php	1101
Function newsletter_add_related	1101
Function newsletter_body2text	1101
Function newsletter_compose_get_dialogdef	1102
Function newsletter_compose_get_dialogdef_delete	1103
Function newsletter_compose_move	1103
Function newsletter_compose_new_sort_order	1103
Function newsletter_compose_save	1104
Function newsletter_compose_show	1104
Function newsletter_compose_show_add	1104
Function newsletter_compose_show_delete	1105
Function newsletter_compose_show_edit	1105
Function newsletter_compose_show_list	1106
Function newsletter_compose_show_list_article	1107
Function newsletter_compose_show_preview	1107
Function newsletter_compose_show_testmail	1108
Function newsletter_compose_toggle	1108
Function newsletter_configuration_dialog_validate	1108
Function newsletter_configuration_get_dialogdef	1109
Function newsletter_configuration_save	1109
Function newsletter_configuration_show	1110
Function newsletter_connect	1110
Function newsletter_content_show	1110
Function newsletter_disconnect	1111
Function newsletter_download_csv	1111
Function newsletter_get_html	1112
Function newsletter_get_html_templates	1113
Function newsletter_publish_dialog_validate	1114
Function newsletter_publish_get_dialogdef	1114

Function newsletter publish prepare issue	1114
Function newsletter publish save	1114
Function newsletter publish show	1115
Function newsletter publish show confirm	1115
Function newsletter publish show preview	1116
Function newsletter publish show testmail	1116
Function newsletter queue get dialogdef	1117
Function newsletter queue save	1117
Function newsletter queue show	1117
Function newsletter save	1118
Function newsletter show edit	1118
Function newsletter strip8859 1	1119
Function newsletter subscriptions get dialogdef	1119
Function newsletter subscriptions get dialogdef delete	1119
Function newsletter subscriptions save	1120
Function newsletter subscriptions show	1120
Function newsletter subscriptions show add	1121
Function newsletter subscriptions show delete	1121
Function newsletter subscriptions show edit	1122
Function newsletter subscriptions show list	1122
Function newsletter subscriptions table	1123
Function newsletter upload csv	1124
Function newsletter upload csv extract	1124
Function newsletter upload csv get dialogdef	1125
Function newsletter upload csv load	1125
Function newsletter upload entries	1125
Function newsletter upload entries get dialogdef	1126
Function newsletter upload entries store	1127
Function newsletter urls2cids	1128
Function newsletter volume number available	1128
newsletter common.php	1130
Define NEWSLETTER_ALERT_SITE	1130
Define NEWSLETTER_CHORE_ADD	1130
Define NEWSLETTER_CHORE_DELETE	1130
Define NEWSLETTER_CHORE_DESELECT	1130
Define NEWSLETTER_CHORE_DOWN	1130
Define NEWSLETTER_CHORE_DOWNLOAD	1130
Define NEWSLETTER_CHORE_EDIT	1130
Define NEWSLETTER_CHORE_LIST	1130
Define NEWSLETTER_CHORE_PREVIEW	1130
Define NEWSLETTER_CHORE_SELECT	1130
Define NEWSLETTER_CHORE_TESTMAIL	1130
Define NEWSLETTER_CHORE_UP	1130
Define NEWSLETTER_CHORE_UPLOAD	1130
Define NEWSLETTER_CONFIRM_ADD	1130
Define NEWSLETTER_CONFIRM_DEL	1130
Define NEWSLETTER_CONFIRM_LEN	1130
Define NEWSLETTER_CONTRIBUTION_POLICY_NO	1130
Define NEWSLETTER_CONTRIBUTION_POLICY_USERS	1130

Define NEWSLETTER CONTRIBUTION POLICY YES	1130
Define NEWSLETTER FULLMAIL ONLY	1130
Define NEWSLETTER FULLMAIL SITE	1130
Define NEWSLETTER SUBSCRIPTION APPROVED	1130
Define NEWSLETTER SUBSCRIPTION BLACKLIST	1130
Define NEWSLETTER SUBSCRIPTION CONFIRMED	1130
Define NEWSLETTER SUBSCRIPTION NEW	1130
Define NEWSLETTER SUBSCRIPTION POLICY NONE	1130
Define NEWSLETTER SUBSCRIPTION POLICY ONE	1130
Define NEWSLETTER SUBSCRIPTION POLICY TWO	1130
Define NEWSLETTER SUBSCRIPTION SKIP	1130
Define NEWSLETTER UNPUBLISHED	1130
Function newsletter_config	1130
Function newsletter consolidate subscribers	1131
Function newsletter dialog_validate	1131
Function newsletter get_emails	1132
Function newsletter queue_run	1132
newsletter_cron.php	1134
Function newsletter_cron	1134
newsletter_install.php	1135
Function newsletter_demodata	1135
Function newsletter_install	1136
Function newsletter_uninstall	1137
Function newsletter_upgrade	1137
newsletter_manifest.php	1138
newsletter_search.php	1139
Function newsletter_search	1139
Function newsletter_search_results	1140
newsletter_view.php	1141
Define NEWSLETTER_DIRNAME	1141
Define NEWSLETTER_REFERENCE	1141
Function newsletter_view	1141
Function newsletter_view_add_navbar	1142
Function newsletter_view_archive	1142
Function newsletter_view_confirm	1143
Function newsletter_view_confirm_get_dialogdef	1143
Function newsletter_view_confirm_process_code	1143
Function newsletter_view_contribution	1143
Function newsletter_view_contribution_form	1144
Function newsletter_view_contribution_get_dialogdef	1144
Function newsletter_view_contribution_preview	1145
Function newsletter_view_contribution_send	1145
Function newsletter_view_contribution_thankyou	1146
Function newsletter_view_home	1146
Function newsletter_view_issue	1146
Function newsletter_view_subscribe	1147
Function newsletter_view_subscribe_get_dialogdef	1147
Function newsletter_view_subscribe_send_code	1147
Function newsletter_view_unsubscribe	1148

Function newsletter_view_unsubscribe_get_dialogdef	1148
Function newsletter_view_unsubscribe_send_code	1148
Package wasmod_redirect Procedural Elements	1151
redirect.php	1151
redirect.php	1152
redirect.php	1153
redirect.php	1154
redirect.php	1155
redirect.php	1156
redirect.php	1157
redirect.php	1158
redirect.php	1159
redirect.php	1160
redirect_admin.php	1161
Function redirect_connect	1161
Function redirect_disconnect	1161
Function redirect_get_dialogdef	1162
Function redirect_save	1162
Function redirect_show_edit	1163
redirect_cron.php	1165
Function redirect_cron	1165
redirect_install.php	1166
Function redirect_demodata	1166
Function redirect_install	1167
Function redirect_uninstall	1167
Function redirect_upgrade	1167
redirect_manifest.php	1169
redirect_search.php	1170
Function redirect_search	1170
redirect_view.php	1171
Function redirect_view	1171
Package wasmod_search Procedural Elements	1174
search_tabledefs.php	1174
search.php	1175
search.php	1176
search_admin.php	1177
Function search_connect	1177
Function search_disconnect	1177
Function search_get_dialogdef	1178
Function search_save	1178
Function search_show_edit	1179
search_cron.php	1181
Function search_cron	1181
search_install.php	1182
Function search_demodata	1182
Function search_install	1183
Function search_uninstall	1183
Function search_upgrade	1183

search_manifest.php	1185
search_search.php	1186
Function search_search	1186
search_view.php	1187
Function search	1187
Function search_areas	1188
Function search_construct_nodelist	1188
Function search_context	1190
Function search_get_places	1191
Function search_get_qwords	1191
Function search_highlight	1192
Function search_module	1192
Function search_navbar	1193
Function search_nodes	1193
Function search_show_form	1194
Function search_simple	1194
Function search_view	1195
Function search_view_dialog_validate	1195
Function search_view_get_config	1196
Function search_view_get_dialogdef	1196
Function search_where	1196

[Package wasmod_sitemap Procedural Elements](#)

sitemap_tabledefs.php	1199
sitemap.php	1200
sitemap.php	1201
sitemap_admin.php	1202
Function sitemap_connect	1202
Function sitemap_disconnect	1203
Function sitemap_get_dialogdef	1203
Function sitemap_save	1203
Function sitemap_show_edit	1204
sitemap_cron.php	1206
Function sitemap_cron	1206
sitemap_install.php	1207
Function sitemap_demodata	1207
Function sitemap_install	1208
Function sitemap_uninstall	1208
Function sitemap_upgrade	1209
sitemap_manifest.php	1210
sitemap_search.php	1211
Function sitemap_search	1211
sitemap_view.php	1212
Function sitemap_tree_walk	1212
Function sitemap_view	1213

[Package wasmod_snapshots Procedural Elements](#)

snapshots_tabledefs.php	1215
snapshots.php	1216
snapshots.php	1217

snapshots_admin.php	1218
Function snapshots_check_path	1218
Function snapshots_connect	1219
Function snapshots_disconnect	1220
Function snapshots_get_dialogdef	1220
Function snapshots_save	1220
Function snapshots_show_edit	1221
snapshots_cron.php	1223
Function snapshots_cron	1223
snapshots_install.php	1224
Function snapshots_demodata	1224
Function snapshots_install	1225
Function snapshots_uninstall	1226
Function snapshots_upgrade	1226
snapshots_manifest.php	1228
snapshots_search.php	1229
Function snapshots_search	1229
snapshots_view.php	1230
Function snapshots_view	1230

[Package wasmod_snapshots Classes](#) 1232

Class SnapshotViewer	1232
Var \$area_id	1232
Var \$default_showtime	1232
Var \$dimension	1232
Var \$domain	1233
Var \$header	1233
Var \$introduction	1233
Var \$module_record	1233
Var \$node_id	1234
Var \$snapshots	1234
Var \$snapshots_path	1234
Var \$theme	1234
Var \$variant	1234
Constructor SnapshotViewer	1235
Method add_snapshot_navbar	1235
Method get_snapshots	1235
Method get_snapshots_configuration	1236
Method images_array	1236
Method javascript_add_configuration	1237
Method javascript_include_once	1237
Method run	1237
Method view_raw	1237
Method view_slideshow	1238
Method view_snapshot	1238
Method view_thumbnails	1238
Class SnapshotViewerInline	1238
Var \$inline_show_height	1239
Var \$inline_show_visible_images	1239
Var \$inline_show_width	1239

Constructor SnapshotViewerInline	1239
Method javascript add inline show	1240
Method noscript add inline show	1240
Method run	1241
Package wastheme_axis Procedural Elements	1243
axis.class.php	1243
axis_install.php	1244
Function axis_demodata	1244
Function axis_get_properties	1245
Function axis_install	1245
Function axis_uninstall	1246
Function axis_upgrade	1246
axis_manifest.php	1248
axis.php	1249
axis.php	1250
Package wastheme_axis Classes	1251
Class ThemeAxis	1251
Method axis_logout	1251
Method axis_printpage	1251
Method get_html	1252
Package wastheme_cornelia Procedural Elements	1254
cornelia_install.php	1254
Function cornelia_demodata	1254
Function cornelia_install	1255
Function cornelia_uninstall	1256
Function cornelia_upgrade	1256
cornelia_manifest.php	1258
cornelia.php	1259
cornelia.php	1260
Package wastheme_cornelia Classes	1261
Class ThemeCornelia	1261
Constructor ThemeCornelia	1261
Method cornelia_get_background_url	1261
Method cornelia_get_sidebar	1262
Method cornelia_get_sidebar_htmlpage	1263
Method cornelia_navigation_index	1263
Method cornelia_sidebar_nodes_modules	1263
Method get_bottomline	1264
Method get_html	1264
Method get_menu	1265
Method get_quicktop	1265
Package wastheme_frugal Procedural Elements	1268
frugal_install.php	1268
Function frugal_demodata	1268
Function frugal_install	1269
Function frugal_uninstall	1269
Function frugal_upgrade	1270

frugal_manifest.php	1271
frugal.php	1272
frugal.php	1273
Package wastheme frugal Classes	1274
Class ThemeFrugal	1274
Package wastheme rosalina Procedural Elements	1276
rosalina.php	1276
rosalina.php	1277
rosalina.class.php	1278
rosalina_install.php	1279
Function rosalina_demodata	1279
Function rosalina_get_properties	1280
Function rosalina_install	1280
Function rosalina_uninstall	1281
Function rosalina_upgrade	1281
rosalina_manifest.php	1283
Package wastheme rosalina Classes	1284
Class ThemeRosalina	1284
Var \$menu_sub	1284
Var \$menu_top	1284
Constructor ThemeRosalina	1284
Method get_html	1285
Method get_logo	1285
Method get_menu_areas	1286
Method get_quickbottom	1286
Method rosalina_get_page_head	1286
Method rosalina_hotspot_map	1287
Method rosalina_hvmenu	1288
Method rosalina_hvmenu_config	1288
Method rosalina_menucount	1288
Method rosalina_menuwidth	1289
Method rosalina_navigation_menu	1289
Method rosalina_show_tree_walk	1289
Package wastheme ruta Procedural Elements	1292
ruta.php	1292
ruta.php	1293
ruta_install.php	1294
Function ruta_demodata	1294
Function ruta_install	1295
Function ruta_uninstall	1295
Function ruta_upgrade	1296
ruta_manifest.php	1297
Package wastheme ruta Classes	1298
Class ThemeRuta	1298
Var \$ruta_header_background	1298
Constructor ThemeRuta	1298
Method alt_show_tree_walk	1299

Method get_altnavigation	1299
Method get_bazaar_style_style	1300
Method get_bottomline	1300
Method get_html	1300
Method get_menu	1300
Method get_menunavigation	1301
Method get_navigation	1301
Method ruta_get_background	1302
Method ruta_get_sidebar	1303
Method ruta_get_sidebar_htmlpage	1303
Method ruta_navigation_index	1303
Method ruta_sidebar_nodes_modules	1304
Method show_menu_js	1304
Package wastheme_schoolyard Procedural Elements	1306
schoolyard.php	1306
schoolyard.php	1307
schoolyard.class.php	1308
schoolyard_install.php	1309
Function schoolyard_demodata	1309
Function schoolyard_get_properties	1310
Function schoolyard_install	1310
Function schoolyard_uninstall	1311
Function schoolyard_upgrade	1311
schoolyard_manifest.php	1313
Package wastheme_schoolyard Classes	1314
Class ThemeSchoolyard	1314
Constructor ThemeSchoolyard	1314
Method get_html	1314
Method schoolyard_get_div_quicktop	1315
Method schoolyard_logout	1315
Method schoolyard_printpage	1315
Package wastheme_sophia Procedural Elements	1317
sophia.php	1317
sophia.php	1318
sophia_install.php	1319
Function sophia_demodata	1319
Function sophia_install	1320
Function sophia_uninstall	1320
Function sophia_upgrade	1321
sophia_manifest.php	1322
Package wastheme_sophia Classes	1323
Class ThemeSophia	1323
Constructor ThemeSophia	1323
Method get_bottomline	1323
Method get_html	1324
Method get_menu	1324
Method get_navigation	1325

Appendices	1326
Appendix A - Class Trees	1327
wascore	1327
default	1330
waslang_ar	1331
waslang_bg	1331
waslang_da	1331
waslang_de	1331
waslang_el	1331
waslang_fa	1331
waslang_en	1331
waslang_es	1331
waslang_fi	1331
waslang_fr	1331
waslang_hi	1331
waslang_hu	1331
waslang_it	1331
waslang_nl	1331
waslang_pl	1331
waslang_pt	1332
waslang_ro	1332
waslang_sh	1332
waslang_ru	1332
waslang_sk	1332
waslang_sv	1332
waslang_tr	1332
waslang_ur	1332
waslang_vi	1332
waslang_zh	1332
waslang_fil	1332
waslang_fry	1332
waslang_pap	1332
wasinstall	1332
wasmod_crew	1332
wasmod_guestbook	1332
wasmod_snapshots	1333
wasmod_htmlpage	1333
wasmod_sitemap	1333
wasmod_mailpage	1333
wasmod_confab	1333
wasmod_mypage	1333
wasmod_search	1333
wasmod_althing	1333
wasmod_newsletter	1333
wasmod_aggregator	1333
wasmod_redirect	1333
wastheme_axis	1333
wastheme_ruta	1334
wastheme_schoolyard	1334

wastheme frugal	1334
wastheme cornelia	1334
wastheme rosalina	1334
wastheme sophia	1335
Appendix B - README/CHANGELOG/INSTALL	1336
FAQ	1337
TODO	1337
LICENSE	1341
README	1341
HISTORY	1342
INSTALL	1342
CHANGES	1343
CREDITS	1387
Appendix D - Todo List	1390

Package wascore Procedural Elements

admin.php

/admin.php - the main entypoint for website maintenance

This is one of the main entry points for Website@School. Other main entry points are /index.php, /cron.php, /file.php and also /program/install.php. Main entry points all define the constant WASENTRY. This is used in various include()ed files to detect break-in attempts.

This is a kickstarter for /program/main_admin.php.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: admin.php,v 1.7 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
require_once dirname(WASENTRY)."/config.php" [line 40]
```

```
require_once $CFG->progdire."/main_admin.php" [line 46]
```

config-example.php

/config-example.php - example of the main configuration file

This file demonstrates and documents the contents of the main configuration file 'config.php'. This file only contains the parameters that are necessary to make a connection to the database and to identify the location of the program directory both in the file system on the webserver and as seen through a web browser from the outside. It is also possible to switch debugging on via the optional \$CFG-debug variable. All other configuration parameters are to be found in the database.

As a rule, the configuration file is generated at installation time. It MUST be called 'config.php' and it MUST reside in the same directory as the main entry points [index.php](#), [admin.php](#), [cron.php](#) and [file.php](#).

Here is an overview of the 11 essential parameters and the 1 optional parameter kept in the global \$CFG object.

- \$CFG->db_type defines the database type.

Currently the only database type supported is 'mysql'. Maybe other databases will be supported in the future.

Default: 'mysql'

- \$CFG->db_server defines the name of the database server.

In the case of 'mysql' the format is 'hostname:port' where 'hostname' is a valid host (default 'localhost') and 'port' is a portnumber or the path to a local socket, e.g. '/var/lib/mysql/mysql.sock'. If the ':' and the portnumber are omitted, the default port 3306 is used.

Default: 'localhost'

- \$CFG->db_username holds the username to use when connecting to the server.

Default: ''

- \$CFG->db_password holds the password to use when connecting to the server.

Default: ''

- \$CFG->db_name holds the name of the database to use.

Default: 'was'

- \$CFG->prefix holds the tablename prefix.

The name of every table is prefixed with this value. This makes it possible to have two or more different instances of Website@School in the same database, simply by using a different prefix for every instance. Using a prefix ending with an underscore makes the resulting table names more readable and it also prevents mis-interpretation of a table name as an SQL keyword.

Default: 'was_'

- `$CFG->dir` is the absolute directory path of 'index.php' and 'config.php'.

The main entry points (index.php, admin.php, etc.) are located in `$CFG->dir` whereas all other program files are located in the directory tree starting in `$CFG->progd`.

Usually the path in `$CFG->dir` is the same as the path to the document root of the webserver.

Examples:

- /var/www/html (Red Hat),
- /home/exemplum/public_html (DirectAdmin),
- /home/httpd/htdocs (Open NA),
- C:\PROGRAM FILES\EASYPHP\WWW (Windows).

Default: '/home/httpd/htdocs'

- `$CFG->www` is the URI which corresponds with the directory `$CFG->dir`.

This URI is of the form `scheme://hostname:port/path`, where - `scheme` is either 'http' or 'https' - `hostname` is the name of the server - `port` is the number of the port the server uses to serve webpages - `path` is a path relative to the document root

Note that the colon followed by a port number are optional; the port number defaults to 80 for http and to 443 for https. Also note that the path is optional. If the path is omitted, the document root of 'hostname' is implied.

Examples:

- `http://www.example.com`
- `https://www.example.com`
- `http://www.example.com:81/web`
- `https://www.example.com:443` (the portnumber is superfluous here)
- `http://www.example.com:80` (the portnumber is superfluous here)

Default: (none)

- `$CFG->progd` is the absolute path to the program directory

The main entry points (index.php, admin.php, etc.) are located in `$CFG->dir` whereas all other program files are located in the directory tree starting in `$CFG->progd`.

Usually this directory is a subdirectory of the document root, or more precise: a subdirectory of `$CFG->dir`. The default name of this subdirectory is 'program'.

Default: '/home/httpd/htdocs/program'

- `$CFG->progwww` is the URI which corresponds with the directory `$CFG->progdir`. This URI is also of the form 'scheme://hostname:port/path', see the explanation for `$CFG->www` above.

As a rule this URI is `$CFG->www` followed by the relative path 'program'.

Important note: If you select different schemes for `$CFG->www` and `$CFG->progwww`, the browser of the website visitor may complain about mixing secure and insecure resources on the same page. It is best to use either 'http' or 'https' and not to mix both.

Default: (none)

- `$CFG->datadir` is the absolute path to a private directory outside the document root. This path points to a directory in which user documents are stored. This directory must be writeable for the user account which runs the webserver (often a user account named 'www-user' or 'www' or 'httpd' or 'apache'). This directory should NOT be accessible from the outside. Note that because of this there is no `$CFG->datawww` which corresponds to `$CFG->datadir`. The data directory should be located outside the document root. All files that need to be served from this directory tree will be served via the program code in 'file.php' in the directory `$CFG->dir`.

Note: some ISPs do not allow you to store data outside the document root. In that case you could use a 'difficult' directory name under the document root, e.g. '/home/httpd/htdocs/d3b07384d113edec49eaa6238ad5ff00', which makes it harder to guess the name and directly access the files in this directory via a browser.

Default: (none)

- `$CFG->debug` is a parameter to switch debugging ON
Via this optional boolean variable debugging can be switched on. If the parameter is not defined, it defaults to FALSE, see [init.php](#).

Default: (variable is not defined)

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: config-example.php,v 1.6 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

cron.php

/cron.php - the main entryptoint for processing cron jobs

This is one of the main entry points for Website@School. Other main entry points are /index.php, /admin.php, /file.php and also /program/install.php. Main entry points all define the constant WASENTRY. This is used in various include()ed files to detect break-in attempts.

This is a kickstarter for /program/main_cron.php.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: cron.php,v 1.7 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

require_once **dirname(WASENTRY)."/config.php"** [*line 40*]

require_once **\$CFG->progdire."/main_cron.php"** [*line 46*]

file.php

/file.php - the main entrypoint for serving files

This is one of the main entry points for Website@School. Other main entry points are [admin.php](#), [cron.php](#), [index.php](#) and also [install.php](#). Main entry points all define the constant `WASENTRY`. This is used in various include()ed files to detect break-in attempts.

This is a kickstarter for `/program/main_file.php`, but first we check for 'maintenance mode': if the file 'maintenance.html' exists in the current directory, we bail out and redirect the visitor to that file.

If we're NOT in maintenance mode, we read the essential configuration parameters from the file 'config.php' in the current directory and continue with `/program/main_index.php` to do the actual work.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: file.php,v 1.8 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

`WASENTRY = __FILE__` [line 40]

Valid entry points define `WASENTRY`; prevents direct access to include()'s.

`require_once $CFG->progdire."/main_file.php"` [line 54]

`require_once dirname(WASENTRY)."/config.php"` [line 45]

index.php

/index.php - the main entryptpoint for website visitors (frontpage)

This is one of the main entry points for Website@School. Other main entry points are [admin.php](#), [cron.php](#), [file.php](#) and also [install.php](#). Main entry points all define the constant WASENTRY. This is used in various include()ed files to detect break-in attempts.

This is a kickstarter for /program/main_index.php, but first we check for 'maintenance mode': if the file 'maintenance.html' exists in the current directory, we bail out and redirect the visitor to that file.

If we're NOT in maintenance mode, we read the essential configuration parameters from the file 'config.php' in the current directory and continue with /program/main_index.php to do the actual work.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: index.php,v 1.8 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

`require_once dirname(WASENTRY)."/config.php" [line 45]`

`require_once $CFG->progdir."/main_index.php" [line 54]`

init.php

/program/init.php - setup database connection, sessions, configuration, etc.

This file is included from one of the main entry points. The following subsystems and global variables are initialised:

- connection to the database
- session handler
- \$CFG
- etc.

This file is included at a fairly early stage in the process. It does not rely on any regular libraries which are include()'ed later on. That is: all relevant libraries (such as [waslib.php](#)) are included when necessary from within the function [initialise\(\)](#).

Note that this file is the place to defined truly global constants because it is always include()'d.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: init.php,v 1.15 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

MAXIMUM_ITERATIONS = 50 *[line 46]*

This global constant defines the maximum number of iterations in database loops (prevent circular reference)

THUMBNAIL_PREFIX = zz_thumb_ *[line 43]*

This global constant is used to specify thumbnail files to be ignored in directory listings

WLOG_ALERT = 1 *[line 52]*

This global constant replaces a similar built-in constant LOG_ALERT which is defined as 1 in win32.h

WLOG_CRIT = 2 *[line 55]*

This global constant replaces a similar built-in constant LOG_CRIT which is erroneously defined as 1 in win32.h

`WLOG_DEBUG = 7` [*line 70*]

This global constant replaces a similar built-in constant `LOG_DEBUG` which is erroneously defined as 6 in `win32.h`

`WLOG_EMERG = 0` [*line 49*]

This global constant replaces a similar built-in constant `LOG_EMERG` which is erroneously defined as 1 in `win32.h`

`WLOG_ERR = 3` [*line 58*]

This global constant replaces a similar built-in constant `LOG_ERR` which is erroneously defined as 4 in `win32.h`

`WLOG_INFO = 6` [*line 67*]

This global constant replaces a similar built-in constant `LOG_INFO` which is defined as 6 in `win32.h`

`WLOG_NOTICE = 5` [*line 64*]

This global constant replaces a similar built-in constant `LOG_NOTICE` which is erroneously defined as 6 in `win32.h`

`WLOG_WARNING = 4` [*line 61*]

This global constant replaces a similar built-in constant `LOG_WARNING` which is erroneously defined as 5 in `win32.h`

double function `diff_microtime($time_start, $time_stop)` [*line 372*]

Function Parameters:

- *string* **`$time_start`** starting time as a string (fractional seconds, space, seconds)
- *string* **`$time_stop`** ending time as a string (fractional seconds, space, seconds)

Calculate the difference between two microtimes

void function `error_exit($bare_condition_code, [$page_title = 'Fatal Error'])` [*line 320*]

Function Parameters:

- *string* **`$bare_condition_code`** the bare condition code to report
- *string* **`$page_title`** the title to show in the generated HTML-page

emergency exit of program in case there is something really, really wrong

This routine outputs a short message and a 'cryptic' condition code and exits the program. It is called when something goes horribly wrong during the early stages of running the program, e.g. the database cannot be opened or there is a version mismatch between the program code (the .php-files) and the database. The complete condition code is the WAS release number followed by a slash followed by the WAS version number followed by a slash and the bare condition code. The message ends with a link to about.html with 'Powered by' or 'Based on', depending on the WAS original flag. Note that we try to show graphics (including logo) but that we switch back to text-only if it is too early, ie. before [waslib.php](#) is included.

Here is an overview of meaning of the condition codes used.

- 010: cannot find config.php, is W@S installed at all?
- 015: cannot find program/main_XXXXX.php, is W@S installed at all?
- 017: cannot calculate wasentry_script_name, are we being tricked?
- 020: configuration error, invalid database type
- 030: cannot connect to database, busy or configuration error?
- 040: error accessing the database, is W@S installed at all?
- 050: version mismatch, update to new version necessary
- 060: magic_quotes_sybase is On
- 070: there is no (default) node available in this (default) area
- 080: there is no area available
- 090: there is no valid theme available

The condition code is numeric because it is easier to report for non-English speaking users than a complicated English sentence. (The language files are not yet loaded when `error_exit()` is called).

If the user was logged in when this fatal error happens, e.g. she requested a protected area, logged in but access was denied anyway, a small postfix is added to the condition code: an asterisk. This indicates that the logged in user was logged out as part of this error exit. This way (hopefully) the session will no longer be available after this error exit.

- **TODO** Q: do we really want to 'leak' a link to the main site?
- **Uses** [WAS_VERSION](#) - indicate internal version in 'cryptic' message
- **Uses** `$WAS_SCRIPT_NAME`
- **Uses** `$USER`
- **Uses** `$CFG`

void function initialise() [*line 76*]

initialise the program, setup database, read configuration, etc.

string function wasentry_script_name(\$full_wasentry_path) [*line 391*]

Function Parameters:

- *string* **\$full_wasentry_path** is the full path of the entry point, e.g. '/home/httpd/htdocs/was/index.php'

determine the name of the executing script (the entry point)

this routine tries to reach consensus about the name of the script that was the entry point. This is not as easy as it sounds.

See [install_script_name\(\)](#) for an exhausting discussion of the issues.

bool/void function was_version_check([\$exit_on_error = TRUE]) [*line 255*]

Function Parameters:

- *bool* **\$exit_on_error** if TRUE, this routine only returns if versions match, if FALSE a mismatch is fatal

check version of PHP-files against version stored in database

this checks the main WAS_VERSION (of files) against \$CFG->version (database). if all is well, we return TRUE indicating both version numbers match. If there is a discrepancy it is logged and depending on parameter \$exit_on_error we either exit altogether OR we return FALSE to indicate the version mismatch.

Typical use is to call this routine near the start as follows (e.g. in [main_index.php](#) or [main_file.php](#)):

```
...
initialise();
was_version_check();
// Still here? Then version is OK
...
```

This forces an exit for the interfaces at the 'visitor' side. For the webmaster it is different: even if the versions do not match, we want to be able to login and do something about it via some sort of upgrade routine, e.g:

```
...
initialise();
if (!was_version_check(FALSE)) {
    do_upgrade();
} else {
    do_regular_admin();
}
...
```


admin.php

/program/languages/en/admin.php - translated messages for /program/admin.php (English)

This file is the 'mother-of-all-languages' file: it is the basis for all other translations. Because there are so many strings to translate, it is easy to lose track of which one is used where, etc. Also, it is sometimes hard to figure out what the purpose of a word or a phrase is without the context.

I try to make that a little easier by adding comments to this (main) language file. These comments will be collected in a separate array called `$comment`. The actual translations and texts end up in an array called `$string`. By using the same key in the `$comment` array, it is possible to add clarification. This clarification can be made visible in the translation tool, helping the translator grasping the purpose and context of the texts to translate.

Note that the order in which the texts appear in this file can also determine the order in which the strings are displayed in the translation tool. The comments 'follow' the strings rather than vice versa. It is not an error if a string doesn't have a comment.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: admin.php,v 1.41 2016/06/15 12:11:41 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/en/loginlib.php - translated messages for login procedure and change password

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: loginlib.php,v 1.8 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/en/was.php - generic translated messages

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: was.php,v 1.11 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/nl/admin.php - translated messages for /program/admin.php
(Dutch)

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: admin.php,v 1.39 2016/06/28 12:34:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/nl/loginlib.php - translated messages for login procedure and change password

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: loginlib.php,v 1.8 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/nl/was.php - generic translated messages (Dutch)

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: was.php,v 1.11 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

accountmanagerlib.php

/program/lib/accountmanagerlib.php - accountmanager (users and groups)

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: accountmanagerlib.php,v 1.7 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

GROUPMANAGER_DIALOG_ADD = 1 *[line 65]*

Distinguish between the various dialogs

GROUPMANAGER_DIALOG_CAPACITY_ADMIN = 14 *[line 70]*

GROUPMANAGER_DIALOG_CAPACITY_INTRANET = 13 *[line 69]*

GROUPMANAGER_DIALOG_CAPACITY_PAGEMANAGER = 15 *[line 71]*

GROUPMANAGER_DIALOG_DELETE = 3 *[line 67]*

GROUPMANAGER_DIALOG_EDIT = 2 *[line 66]*

TASK_ACCOUNTS = overview *[line 28]*

default selection for account manager: show introduction + links to users and groups

TASK_GROUPS = groups *[line 49]*

TASK_GROUP* relate to plain groups

TASK_GROUP_ADD = groupadd *[line 50]*

TASK_GROUP_CAPACITY_ADMIN = capacityadmin *[line 60]*

TASK_GROUP_CAPACITY_INTRANET = capacityintranet *[line 58]*

TASK_GROUP_CAPACITY_MODULE = capacitymodule *[line 59]*

TASK_GROUP_CAPACITY_OVERVIEW = capacityoverview *[line 57]*

TASK_GROUP_CAPACITY_* relate to group-capacity-combinations

TASK_GROUP_CAPACITY_PAGEMANAGER = capacitypagemanager *[line 61]*

TASK_GROUP_CAPACITY_SAVE = capacitysave *[line 62]*

TASK_GROUP_DELETE = groupdelete *[line 51]*

TASK_GROUP_EDIT = groupedit *[line 52]*

TASK_GROUP_SAVE = groupsave *[line 53]*

TASK_GROUP_SAVE_NEW = groupsavenew *[line 54]*

TASK_USERS = users *[line 31]*

TASK_USER* relate to user accounts

```

TASK_USER_ADD = useradd [line 32]
TASK_USER_ADMIN = useradmin [line 42]
TASK_USER_ADVANCED = useradvanced [line 35]
TASK_USER_DELETE = userdelete [line 33]
TASK_USER_EDIT = useredit [line 34]
TASK_USER_GROUPADD = usergroupadd [line 37]
TASK_USER_GROUPDELETE = usergroupdelete [line 38]
TASK_USER_GROUPS = usergroups [line 36]
TASK_USER_GROUPSAVE = usergroupsave [line 39]
TASK_USER_INTRANET = userintranet [line 40]
TASK_USER_MODULE = usermodule [line 41]
TASK_USER_PAGEMANAGER = userpagemanager [line 43]
TASK_USER_SAVE = usersave [line 45]
TASK_USER_SAVE_NEW = usersavenew [line 46]
TASK_USER_TREEVIEW = usertreeview [line 44]
USERMANAGER_DIALOG_ADD = 21 [line 73]
USERMANAGER_DIALOG_ADMIN = 34 [line 79]
USERMANAGER_DIALOG_DELETE = 23 [line 75]
USERMANAGER_DIALOG_EDIT = 22 [line 74]
USERMANAGER_DIALOG_INTRANET = 33 [line 78]
USERMANAGER_DIALOG_PAGEMANAGER = 35 [line 80]
void function job_accountmanager(&$output) [line 96]

```

Function Parameters:

- *object* **&\$output** collects the html output

main entry point for accountmanager (called from admin.php)

this routing dispatches the tasks. If a specified task is not recognised, the default task TASK_ACCOUNTS_OVERVIEW is executed. Note that the User Manager and the Group Manager are heavily interconnected. Therefore we use 1 common set of tasks and distinguish between both managers via sets of tasks, e.g. TASK_USER* point to the user manager where TASK_GROUP* lead to the group manager.

```
void function show_accounts_intro(&$output) [line 173]
```

Function Parameters:

- *object* **&\$output** collects the html output

display an introductory text for the account manager + menu

```
void function show_accounts_menu(&$output, [$current_task = NULL]) [line 223]
```

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **\$current_task** indicate the current menu selection (if any)

display the account manager menu

aclmanager.class.php

/program/lib/aclmanager.class.php - dealing with access control lists

This file defines a class for dealing (edit+save but not create or delete) with lists of access control parameters. The main purpose is to allow easy editing of the many many permission bitmaps that are possible for both users and groups.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: aclmanager.class.php,v 1.13 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

ACL_LEVEL_AREA = 3 *[line 55]*

limit available role options to pages, sections and areas (used in pagemanager permissions)

ACL_LEVEL_NONE = 0 *[line 46]*

limit available role options to 'none' and 'guru' (used in pagemanager permissions)

ACL_LEVEL_PAGE = 1 *[line 49]*

limit available role options to pages (used in pagemanager permissions)

ACL_LEVEL_SECTION = 2 *[line 52]*

limit available role options to pages and sections (used in pagemanager permissions)

ACL_LEVEL_SITE = 4 *[line 58]*

no limit on available role options (used in pagemanager permissions)

ACL_TYPE_ADMIN = 2 *[line 36]*

acl for administrator permissions

ACL_TYPE_INTRANET = 1 *[line 33]*

acl for intranet permissions

ACL_TYPE_MODULE = 4 *[line 42]*

acl for individual module permissions

ACL_TYPE_PAGEMANAGER = 3 *[line 39]*

acl for pagemanager permissions

alertmanager.class.php

/program/lib/alertmanager.class.php - manage alerts for changes in areas/nodes

This file defines a class for managing alerts (add, edit, delete, view) and underlying rules (add, delete).

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: alertmanager.class.php,v 1.1 2016/04/26 09:33:19 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
ALERTMANAGER_CHORE_ADD = add [line 31]
ALERTMANAGER_CHORE_DELETE = delete [line 33]
ALERTMANAGER_CHORE_EDIT = edit [line 32]
ALERTMANAGER_CHORE_RULE_ADD = addrule [line 35]
ALERTMANAGER_CHORE_RULE_DELETE = deleterule [line 37]
ALERTMANAGER_CHORE_RULE_EDIT = editrule [line 36]
ALERTMANAGER_CHORE_RULE_SAVE = saverule [line 38]
ALERTMANAGER_CHORE_SAVE = save [line 34]
ALERTMANAGER_CHORE_VIEW = view [line 30]
```

areamanager.class.php

/program/lib/areamanager.class.php - taking care of area management

This file defines a class for managing areas (add, edit, delete, view).

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: areamanager.class.php,v 1.18 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
AREAMANAGER_CHORE_ADD = add [line 30]
AREAMANAGER_CHORE_DELETE = delete [line 31]
AREAMANAGER_CHORE_EDIT = edit [line 32]
AREAMANAGER_CHORE_EDIT_THEME = edittheme [line 33]
AREAMANAGER_CHORE_RESET_THEME = resettheme [line 34]
AREAMANAGER_CHORE_SAVE = save [line 36]
AREAMANAGER_CHORE_SAVE_NEW = savenew [line 35]
AREAMANAGER_CHORE_SET_DEFAULT = setdefault [line 37]
AREAMANAGER_CHORE_VIEW = view [line 29]
AREAMANAGER_DIALOG_ADD = 1 [line 39]
AREAMANAGER_DIALOG_DELETE = 2 [line 40]
AREAMANAGER_DIALOG_EDIT = 3 [line 41]
AREAMANAGER_DIALOG_EDIT_THEME = 4 [line 42]
AREAMANAGER_DIALOG_RESET_THEME = 5 [line 43]
```

configassistant.class.php

/program/lib/configassistant.class.php - dealing with lists of configuration parameters

This file defines a class for dealing (edit+save but not create or delete) with lists of configuration parameters. The main purpose is to allow easy editing of configuration of parts of the system, including the main program configuration.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: configassistant.class.php,v 1.12 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

configurationmanagerlib.php

/program/lib/configurationmanagerlib.php - configurationmanager

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: configurationmanagerlib.php,v 1.10 2016/04/26 09:33:19 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
CHORE_SAVE = save [line 34]
TASK_ALERTS = alerts [line 32]
TASK_AREAS = areas [line 30]
TASK_CONFIGURATION_INTRO = intro [line 29]
TASK_SITE = site [line 31]
void function job_configurationmanager(&$output) [line 46]
```

Function Parameters:

- *object* **&\$output** collects the html output

main entry point for configurationmanager (called from /program/main_admin.php)

this routine dispatches the tasks, If the specified task is not recognised, the default task TASK_CONFIGURATION_INTRO is executed.

```
void function process_task_site(&$output) [line 167]
```

Function Parameters:

- *object* **&\$output** collects the html output

handle the editing/saving of the main configuration information

this routine handles editing of the main configuration parameters. It either displays the edit dialog or saves the modified data and shows the configuration manager introduction screen.

Note that we do NOT try to redirect the user via a header() after a succesful save. It would be

handy because this particular save action may have had impact on the global configuration, which is already read at this point. By redirecting we would make a fresh start, with the new parameters. However, we lose the easy ability to tell the user that the data was saved (via `$output->add_message()`). So, either no feedback or obsolete global config in core. Hmmmm. I settle for the feedback and the 'wrong' settings.

- **Uses** ConfigAssistant()

void function show_configuration_intro(&\$output) [line 102]

Function Parameters:

- *object* **&\$output** collects the html output

display an introductory text for the configuration manager + menu

void function show_configuration_menu(&\$output, [\$current_task = NULL]) [line 114]

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **\$current_task** indicate the current menu selection (if any)

display the configuration manager menu

database.lib.php

/program/lib/database/database.lib.php - database factory and database access routines

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: database.lib.php,v 1.7 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool/object function database_factory(\$prefix, [\$db_type = 'mysql'], [\$debug = FALSE]) [*line 64*]

Function Parameters:

- *string* **\$prefix** the tablename prefix
- *string* **\$db_type** (optional) which database to use, default 'mysql'
- *bool* **\$debug** if TRUE extra information is displayed (handy for debugging the code)

manufacture a database object

This loads (includes) a specific database access class based on the parameter \$db_type. Currently 'mysql' is the only option, but support for PostgreSQL or other databases could be added in the future, see the code that is commented out

Because Website@School is not meant to be an 'enterprisy application', I decided against using an abstract class that would be extended by a specific driver class which would be instantiated via yet another factory type class; I'd like to keep this as simple as possible while retaining the necessary flexibility (and the option to add support for other databases). Toolkits like Adodb seem overkill for this application program.

This routine is called at a fairly early stage in the process. It does not rely on any regular libraries which may be include()'ed lateron. If no valid database type is specified, the function returns FALSE, otherwise a database object is returned.

Note that I did not use a singleton because I think that that pattern is simply a fancy word for a global variable. YMMV.

Note: This file can be safely included from the [install.php](#) script, allowing for database-

manipulations via this abstraction layer rather than directly going to the database. There are no dependencies on other include()'s other than the actual database class files such as mysql.class.php. Also, this file does not rely on the global variable \$CFG, which is also very convenient in the installer (where no CFG is available).

- **TODO** perhaps add postgresql in a future version

void function db_bool_is(\$value, \$variable_to_check) [line 385]

Function Parameters:

- *bool/mixed* **\$value** value to test for, could be TRUE, FALSE or anything else
- *mixed* **\$variable_to_check** the value of the variable to check

check boolean field in a database-independent way

Various databases have different ways to indicate TRUE or FALSE in boolean type of fields. MySQL uses a tinyint(1) with values NULL, 0 and 1. PostgreSQL uses a lowercase 't' or 'f' etc. We already have two database-specific definitions for TRUE and FALSE: SQL_TRUE and SQL_FALSE. This routine 'converts' the database-specific boolean values back to a form that is useable in PHP. This routine is able to test for either TRUE or FALSE. Any other value is returned as NULL.

```
Typical use: $user = db_select_single_record('users','is_active','user_id = 13');
if (db_bool_is(TRUE,$user['is_active'])) {
    ...
}
```

bool/int function db_delete(\$tablename, [\$where = ""]) [line 341]

Function Parameters:

- *string* **\$tablename** the name of the table to delete from (without prefix)
- *mixed* **\$where** a single clause or an array with fieldnames => values ((without the WHERE keyword)

delete zero or more rows in a table

- Uses [db_delete_sql\(\)](#)

string function db_delete_sql(\$tablename, [\$where = ""]) [*line 353*]

Function Parameters:

- *string* **\$tablename** the name of the table to delete from (without prefix)
- *mixed* **\$where** a single clause or an array with fieldnames => values ((without the WHERE keyword)

generate SQL to delete zero or more rows in a table

- Usedby [db_delete\(\)](#)

string function db_errormessage() [*line 463*]

retrieve the latest database error from \$DB

- Uses \$DB;

string/mixed function db_escape_and_quote(\$value) [*line 282*]

Function Parameters:

- *mixed* **\$value** string, boolean, null or other value to escape and quote

conditionally quote and escape values for use with a database table

If \$value is a string, it is escaped and single quotes are added at begin and end. If \$value is a boolean, it is converted into the correct value for the database using SQL_FALSE/SQL_TRUE If \$value is NULL, it is converted into the string 'NULL' (without

quotes) Otherwise the value is not changed.

- See <http://xkcd.com/327>
- Usedby [db_select_sql\(\)](#)
- Uses \$DB
- Usedby [db_insert_into_sql\(\)](#)

bool function db_insert_into(\$tablename, \$fields) [line 125]

Function Parameters:

- *string* **\$tablename** the name of the table to insert into (without prefix)
- *array* **\$fields** an associative array with fieldnames and fieldvalues

execute the necessary SQL-code for an INSERT INTO statement

This excutes the SQL-statement created by [db_insert_into_sql\(\)](#).

- Uses \$DB

bool function db_insert_into_and_get_id(\$tablename, \$fields, [\$key_fieldname = ""]) [line 142]

Function Parameters:

- *string* **\$tablename** the name of the table to insert into (without prefix)
- *array* **\$fields** an associative array with fieldnames and fieldvalues
- *string* **\$key_fieldname** the name of the field that holds the primary key/serial

execute the necessary SQL-code for an INSERT INTO statement and return the last_insert_id

This executes the SQL-statement created by [db_insert_into_sql\(\)](#). If all goes well, the value of the last inserted id is returned.

- **Uses** `$DB`

string function `db_insert_into_sql($tablename, $fields)` [line 100]

Function Parameters:

- *string* **\$tablename** the name of the table to insert into (without prefix)
- *array* **\$fields** an associative array with fieldnames and fieldvalues

generate the necessary SQL-code for an INSERT INTO statement

Construct an SQL-statement that inserts data into the specified table. This routine takes care of properly escaping strings and also handles the addition of the table prefix

- **Uses** [db_escape_and_quote\(\)](#)

int|bool function `db_last_insert_id([$tablename = ''], [$fieldname = ''])` [line 414]

Function Parameters:

- *string* **\$tablename** name of the table (without prefix) in which a record was inserted
- *string* **\$fieldname** name of the serial field to examine

wrapper for DB->last_insert_id()

This calls `$DB->last_insert_id()` in a way that should be compatible with a future PostgreSQL database class. Note that MySQL doesn't care about this. You can get away with leaving table and field parameters empty (as is the default), but for compatibility and documentation purposes you should use the correct values.

Typical use: `db_insert_into('users',$fields_array);` `$user_id` =

```
db_last_insert_id('users','user_id');
```

- **Uses** `$DB`

bool/array function `db_select_all_records($tablename, $fields, [$where = "], [$order = "], [$keyfield = "], [$limit = "], [$offset = "])` *[line 256]*

Function Parameters:

- *string* **\$tablename** name of the table to select from (without prefix)
- *mixed* **\$fields** fieldname or array with fieldnames to select
- *mixed* **\$where** a single clause or an array with fieldnames => values ((without the WHERE keyword)
- *mixed* **\$order** fieldname or array with fieldnames to determine sort order (without ORDER BY keyword)
- *string* **\$keyfield** field to use as the key in the returned array or empty for 0-based numeric array key
- *int* **\$limit** the maximum number of records to retrieve
- *int* **\$offset** the number of records to skip initially

fetch all selected records from the database in one array

- **Uses** [db_select_sql\(\)](#)

bool/array function `db_select_single_record($tablename, $fields, [$where = "], [$order = "])` *[line 227]*

Function Parameters:

- *string* **\$tablename** name of the table to select from (without prefix)
- *mixed* **\$fields** fieldname or array with fieldnames to select

- *mixed* **\$where** a single clause or an array with fieldnames => values ((without the WHERE keyword)
- *mixed* **\$order** fieldname or array with fieldnames to determine sort order (without ORDER BY keyword)

fetch a single record from the database

- Uses [db_select_sql\(\)](#)

string function db_select_sql(\$tablename, \$fields, [\$where = ''], [\$order = '']) [*line 183*]

Function Parameters:

- *string* **\$tablename** name of the table to select from (without prefix)
- *mixed* **\$fields** fieldname or array with fieldnames to select
- *mixed* **\$where** a single clause or an array with fieldnames => values ((without the WHERE keyword)
- *mixed* **\$order** fieldname or array with fieldnames to determine sort order (without ORDER BY keyword)

generate the necessary SQL-code for a simple SELECT statement

Construct an SQL-statement of the form: SELECT field_list FROM table WHERE where_expression ORDER BY orderby_list

The parameter \$fields can be either a simple string, indicating a single field or an array when more fields are to be selected

The optional parameter \$where is either a simple string with an appropriate expression (without the keyword WHERE) or an array with fieldname/value-pairs. In the latter case the clauses fieldname=value are AND'ed together. If the specified values are string-like, they are properly quoted. Boolean values are treated properly too.

The optional parameter \$order is either a simple string with an appropriate list or expression (without the keyword ORDER BY) or an array with fieldnames which will be used to create a comma-delimited string.

Examples:

1. `db_select_sql('areas','title')` yields `'SELECT title FROM was_areas'`
2. `db_select_sql('areas',array('title','theme_id'),array('is_visible' => TRUE),'sort_order')` yields `'SELECT title,theme_id FROM was_areas WHERE is_visible = 1 ORDER BY sort_order'` (if `SQL_TRUE` is `'1'`)

- Usedby [db_select_all_records\(\)](#)
- Usedby [db_select_single_record\(\)](#)
- Uses [db_escape_and_quote\(\)](#)

bool/int function `db_update($tablename, $fields, [$where = ''])` [*line 305*]

Function Parameters:

- *string* **\$tablename** the name of the table to update (without prefix)
- *array* **\$fields** an associative array with fieldnames and fieldvalues
- *mixed* **\$where** a single clause or an array with fieldnames => values ((without the WHERE keyword))

update one or more fields in a table

- Uses [db_update_sql\(\)](#)

string function `db_update_sql($tablename, $fields, [$where = ''])` [*line 319*]

Function Parameters:

- *string* **\$tablename** the name of the table to update (without prefix)
- *array* **\$fields** an associative array with fieldnames and fieldvalues
- *mixed* **\$where** a single clause or an array with fieldnames => values ((without the WHERE keyword))

generate sql to update one or more fields in a table

- Usedby [db_update\(\)](#)
- Uses \$DB

string function db_where_clause([\$where = ""]) [*line 436*]

Function Parameters:

- *mixed* **\$where** a single clause or an array with fieldnames => values ((without the WHERE keyword)

construct a where clause from string/array, including the word WHERE

this constructs a where clause including the word 'WHERE' based on the string or array \$where.

The optional parameter \$where is either a simple string with an appropriate expression (without the keyword WHERE) or an array with fieldname/value-pairs. In the latter case the clauses fieldname=value are AND'ed together. If the specified values are string-like, they are properly quoted. Boolean values are treated properly too. NULL-values yield a standard 'IS NULL' type of expression.

mysql.class.php

/program/lib/database/mysql.class.php - access to mysql via database class

This file is included from the poor mans database factory in [databaselib.php](#). It provides all necessary functionality to use MySQL as the underlying database server.

This file defines two classes: DatabaseMysql and DatabaseMysqlResult. Typical usage would be:

```
$DB = new Database($table_prefix);           // once at program start
$DB->connect($host,$usr,$pwd,$dbname);        // once at program start

$sql = "SELECT * FROM {$DB->prefix}foo";      // select all rows from table foo,
$DBResult = $DB->query($sql);
$all_rows = $DBResult->fetch_all_assoc()      // store everything in an array,
$DBResult->close();                          // and free the memory

...                                          // do something with $all_rows

$DB->close();                               // once at program end
```

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: mysql.class.php,v 1.12 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

mysqli.class.php

/program/lib/database/mysqli.class.php - access to mysql via database class

This file is included from the poor mans database factory in [databaselib.php](#). It provides all necessary functionality to use MySQL as the underlying database server via the mysqli interface.

This file defines two classes: DatabaseMysqli and DatabaseMysqliResult. Typical usage would be:

```
$DB = new DatabaseMysqli($table_prefix); // once at program start
$DB->connect($host,$usr,$pwd,$dbname); // once at program start

$sql = "SELECT * FROM {$DB->prefix}foo"; // select all rows from table foo,
$DBResult = $DB->query($sql);
$all_rows = $DBResult->fetch_all_assoc() // store everything in an array,
$DBResult->close(); // and free the memory

... // do something with $all_rows

$DB->close(); // once at program end
```

Note that this class is very similar to [mysqli.class.php](#).

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: mysqli.class.php,v 1.2 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

SQL_FALSE = 0 [line 50]

this circumvents the sub-optimal implementation of booleans in MySQL

SQL_TRUE = 1 [line 53]

this circumvents the sub-optimal implementation of booleans in MySQL

dbsessionlib.php

/program/lib/dbsessionlib.php - functions to keep PHP-sessions in the database

This file provides the functions to handle sessions via the database rather than via files or other standard PHP-mechanisms.

Useful information about storing sessions can be found in these user comments:
<http://php.net/manual/en/function.session-set-save-handler.php>

Important issues:

- there is an issue with writing session data when using a database class because the database class is already destroyed when PHP tries to write the session data. Workaround: call `session_commit()` (alias for `session_write_close()`) near the end of the script.
- there is a security risk when the script simply accepts any session id that is presented via a cookie; a session should exist/be created before data is written and the session key should have been created in a previous call and thus be present in the database
- different versions of PHP handle callback parameters differently where object methods are involved. The most generic way to work around these incompatibilities is to use global functions for open, close, etc. rather than methods in some session class.
- session keys should be sanitised before manipulating the database in order to prevent an SQL injection.
- `session.auto_start` may make it impossible to substitute our own session handlers if it is set in `php.ini`
- how about locking a session?

There is a difference between the maximum session duration and the session time out. A session could last for 'duration' seconds but only if there is still activity at least every timeout seconds. There should be a maximum session lifetime of say 24 hours. There also should be a timeout of say 60 minutes.

Sessions are stored in a table called 'sessions'. This table is defined as follows: session_id serial

session_key varchar(172)

session_data longtext

user_id int unsigned

user_information varchar(255)

ctime datetime

atime datetime

primary key(session_id)

foreign key(user_id) references users(user_id)

unique index(session_key)

Note: the size of the session_key was reduced from 255 to 172 after version 2011051100 to prevent database problems (see `update_core_2011092100()`).

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: dbsessionlib.php,v 1.11 2016/05/18 13:59:49 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **Usedby** [admin_continue_session\(\)](#)
- **License** [GNU AGPLv3+Additional Terms](#)

bool function dbsession_close() [*line 217*]

'close' a session that was opened with dbsession_open() before

Since this function has no way to tell _which_ session should be closed, it is utterly useless (but it has to exist to satisfy session_set_save_handler()) The function [dbsession_open\(\)](#) has the same uselessness, so they are a perfect pair.

- **TODO** should we do something with locking the session record from dbsession_open() until dbsession_close()? For now, the session record is not locked in any way, so the latest call gets to keep its changes Mmmm....

bool/string function dbsession_create(\$user_id, [\$user_information = ""]) [*line 128*]

Function Parameters:

- *int* **\$user_id** link to the users table, identifies the user that started the session
- *string* **\$user_information** (optional) auxiliary information about the user, e.g. the IP-address

create a new session in the session table, return the unique sessionkey

this creates generates a new unique session key and stores it in a new record in the sessions table. Additional information is recorded in the new record too: the user_id and auxiliary information. This information makes that a session can always be linked to a particular user (which is handy when dealing with locked pages, etc.). This routine attempts to create a unique session key a number of times. If it doesn't work out, the routine returns

FALSE.

the optional parameter `$user_information` can be used to store additional information about this user, e.g. the IP-address. This is useful for generating messages like 'Node xxx is currently locked by user YYYY logged in from ZZZZ'.

Note that the generation of a unique session key is salted with both the main url of this website and the special salt that was recorded once during installation time. Also, pseudo-random data is added via `rand()`. Hopefully this will be hard to guess, even though we use `md5()` to condense this (semi-)random information into only 128 bits.

- **TODO** should we also record the IP-address of the user in the session record? In a way this is a case of information leak, even though it is only between authenticated users. Mmmm...
- **Uses** `$DB`
- **Uses** `$CFG`

bool function dbsession_destroy(\$session_key) [line 270]

Function Parameters:

- *string* **`$session_key`** the unique session_key that identifies the session

remove a session record from the sessions table (it should still exist)

remove the specified record from the sessions table. it is an error if the record does not exist.

bool function dbsession_exists(\$session_key) [line 162]

Function Parameters:

- *string* **`$session_key`** the unique session_key that identifies the session

check to see if `$session_key` exists in the session table

This checks the existence of a session in the sessions table. Session keys are only generated from [dbsession_create\(\)](#). This prevents us accepting spurious session keys via a manipulated cookie. If the session key does not exist, the call fails and FALSE is returned.

bool function dbsession_expire(\$max_lifetime) [*line 316*]

Function Parameters:

- *int* **\$max_lifetime** the lifetime value to automatically expire sessions (in seconds)

remove all sessions that were created more than \$max_life seconds ago

not only are sessions terminated when there is no more activity for \$time_out seconds (@see dbsession_garbage_collection()) but also the total lifetime of a session is limited to \$life_time seconds. This routine is not part of the required session handlers but it can be called periodically (@see cron.php).

- Uses [dbsession_remove_obsolete_sessions\(\)](#)

bool function dbsession_garbage_collection(\$time_out) [*line 294*]

Function Parameters:

- *int* **\$time_out** the time-out value to automatically expire sessions (in seconds)

remove all sessions that are last accessed more than \$time_out seconds ago, maybe even more

we have our own session expire limit in \$CFG->session_expiry and we overrule the \$ime_out value here (ie. we ignore the php.ini setting) A session times out after \$CFG->session_expiry seconds of inactivity. Also a session is terminated twice that amount of time after the start of the session. Both checks are performed here.

- Usedby [was_login\(\)](#)
- Uses [dbsession_remove_obsolete_sessions\(\)](#)

int|bool function dbsession_get_session_id(\$session_key) [*line 176*]

Function Parameters:

- *string* **\$session_key** the unique session_key that identifies the session

retrieve the session_id (pkey) that corresponds with session_key

this is very similar to [dbsession_exists\(\)](#). This routine returns the actual session_id integer, whereas dbsession_exists() only returns TRUE.

bool function dbsession_open(\$save_path, \$session_name) [line 198]

Function Parameters:

- *string* **\$save_path** (unused) pathname relevant for file based session handler
- *string* **\$session_name** (unused) the non-unique session_name that identifies the cookie in the user's browser

'open' a session

this 'opens' a session. note that this function is unable to identify the session because it is only presented with

- the \$save_path (which is relevant only with file-based session handlers)
- the \$session_name (which is the _name_ of the session, but not the _token_)

there is no way to let this function do anything useful, so it boils down to a dummy always returning TRUE. The function [dbsession_close\(\)](#) has the same uselessness, so they are a perfect pair.

string function dbsession_read(\$session_key) [line 227]

Function Parameters:

- *string* **\$session_key** the unique session_key that identifies the session

read the (serialised) session data from the database

bool function dbsession_remove_obsolete_sessions(\$seconds, \$time_field) [line 340]

Function Parameters:

- *int* **\$seconds** the period of time after which the session is obsolete

- *string* **\$time_field** the field to use for time comparisons: either 'atime' or 'ctime'

workhorse for removing obsolete sessions from the database

this logs and subsequently removes obsolete sessions from the sessions table It is a workhorse function for both [dbsession_garbage_collection\(\)](#) and [dbsession_expire\(\)](#).

Session records are removed when the \$time_field in the sessions table contains a date/time that is older than \$seconds seconds ago. Before the records are removed, we retrieve them and log pertinent information from each one via logger(), for future reference.

Note that we try to continue with deleting records, even if the logging appears to have generated errors.

- **TODO** this routine should not need to know about nodes and stuff. Mmmm... remove FK constraint?
- **Usedby** [dbsession_expire\(\)](#)
- **Usedby** [dbsession_garbage_collection\(\)](#)

bool function dbsession_setup(\$session_name) [line 88]

Function Parameters:

- *string* **\$session_name** the name of the session (usually 'PHPSESSID' in generic PHP-applications)

setup database based handlers for session management

this is basically shorthand for session_set_save_handler() this routine replaces the existing session handlers with the handlers specified below in this file.

- **Usedby** [was_login\(\)](#)
- **Usedby** [was_logout\(\)](#)

bool function dbsession_write(\$session_key, \$session_data) [*line 247*]

Function Parameters:

- *string* **\$session_key** the unique session_key that identifies the session
- *string* **\$session_data** the string with (serialised) session variables

write the (serialised) data to the database

/program/lib/dialoglib.php - useful functions for manipulating dialogs

This file provides various utility routines for creating and validating user dialogs.

A dialog is a collection of input elements grouped together in a form. An input element (or field) has at least a type (e.g. F_ALPHANUMERIC or F_DATETIME) and a name (e.g. 'title' or 'expiry') and optionally one or more of the other possible properties. The name of an input element uniquely identifies the field in the dialog; it can be used to retrieve the data entered by the user from the global \$_POST array.

Note that it is also possible to choose a field name ending in '[]'. This eventually yields an array within the `$_POST` array. This is used in the multiple file upload feature.

A dialog can be defined via a 0-based or associative array, where every array element is separate input element. Each of the input elements is in itself an associative array with property-value-pairs. The recognition of a property depends on the type, e.g. the number of decimals is irrelevant for a string-type input element.

Here is an overview of properties and field types to which they apply. An 'x' means required, an 'o' means optional and a '-' means don't care.

type

```

| name
| | value
| | | rows
| | | | columns
| | | | | minlength
| | | | | maxlength
| | | | | decimals
| | | | | | minvalue
| | | | | | maxvalue
| | | | | | | options
| | | | | | | label
| | | | | | | accesskey
| | | | | | | alt
| | | | | | | class
| | | | | | | | multiple
| | | | | | | | viewonly
| | | | | | | | | tabindex
| | | | | | | | | title
| | | | | | | | | hidden
| | | | | | | | | | autocomplete [1]
| | | | | | | | | | id
| | | | | | | | | | |

```

```

F_ALPHANUMERIC  xx000  00---  -0000  -0000  00
F_INTEGER      xx0-0  00-00  -0000  -0000  00
F_REAL         xx0-0  00000  -0000  -0000  00

```

```

F_DATE      x x 0 - 0 0 0 - 0 0 - 0 0 0 0 - 0 0 0 0 0 0
F_TIME      x x 0 - 0 0 0 - 0 0 - 0 0 0 0 - 0 0 0 0 0 0
F_DATETIME  x x 0 - 0 0 0 - 0 0 - 0 0 0 0 - 0 0 0 0 0 0
F_PASSWORD  x x 0 - 0 0 0 - - - - 0 0 0 0 - 0 0 0 0 0 0
F_CHECKBOX  x x 0 - - - - - x 0 0 0 0 - 0 0 0 0 - 0
F_LISTBOX   x x 0 0 - - - - - x 0 0 0 0 - 0 0 0 0 - 0
F_RADIO     x x 0 - - - - - x 0 0 0 0 - 0 0 0 0 - 0
F_FILE      x x 0 - 0 - - - - - 0 0 0 0 0 0 0 0 - - 0
F_SUBMIT    x x x - - - - - - - 0 0 0 - 0 0 0 - - 0
F_RICHTEXT  x x 0 0 0 0 0 0 - - - - 0 0 0 0 - 0 0 0 0 - 0
F_CSRFTOKEN x x x - - - - - - - - - - - - x - -

```

There are two more properties: 'errors' and 'error_messages'. These properties can be set whenever the validation of the input yields an error. If all is well, 'errors' is either not set or equal to 0.

[1] Note that autocomplete is a non-standard function that doesn't work in every browser.

Here is a description of the various properties.

- type:
the type of the input element, one of the F_* constants defined at the top of this file.
- name:
the name to uniquely identify the input element
- value:
the current value of the element, this value needs to be displayed in the dialog
- rows:
the number of text rows for this element. This applies only to generic text fields (alphanumerics) and listboxes.
- columns:
the number of characters to show in the input element. This applies only to text inputs (not lists or checkboxes).
- minlength:
the minimum number of characters that need to be entered by the user. If 0, the field can be left empty.
- maxlength:
the maximum number of characters that can be entered by the user. Note that this number is no necessarily the same as the number of columns to display (or even the product of columns and rows).

- decimals:
the number of decimals in the fraction of real numbers
- minvalue:
the minimum value that needs to be entered. This applies to numeric fields (integer, real) and also to dates and times.
- maxvalue:
the maximum value that can be entered. This applies to numeric fields (integer, real) and also to dates and times.
- options:
this is a list (array) of key-value-pairs that identify the allowable values for a listbox or radio buttons. The key is used as the fields value and the value is the descriptive title of the option.
- label:
this is the text that is displayed before the input element. It is used to indicate the purpose of the input element.
- accesskey:
this is a single letter that can be used to access the element via the keyboard by using a key combination like [Alt-A].
- alt:
an alternative text that describes the element (accessibility)
- class:
this identifies the class(es) that need to be associated with this element. This allows for changing the style of the element.
- viewonly:
indicates that this element is not to be changed by the user but that it can be displayed nevertheless
- tabindex:
a number that indicates the order in which fields area accessed when using the [Tab] key to move to the next element.
- title:
a descriptive title that is displayed when the mouse is hovering over the element
- hidden:
identifies a field that should be part of the dialog, but not visible to the user. Note that by specifying a hidden list, it is possible to validate the value of the hidden input against the list of acceptable values once it returns from the user. However, one

should never trust the value of a field that is sent in 'hidden' form, because after all it is still user input, no matter what.

- **autocomplete:**
this is a browser-specific attempt to prevent the browser from remembering data entered in a text input field. This does not work in every browser so it is a 'best effort' feature at the most.
- **id**
a unique (within the document) identifier for this input element. This id is used to link a label to an input. The id should start with a letter. The corresponding label is identified by appending the string '_label' to the id.
- **errors:**
the number of errors encountered after validating the value of this element
- **error_messages:**
an array of messages identifying the problems encountered with this element during validation

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: dialoglib.php,v 1.20 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
ATTR_CLASS_ERROR = error [line 215]
ATTR_CLASS_VIEWONLY = viewonly [line 216]
BUTTON_CANCEL = cancel [line 219]
BUTTON_DELETE = delete [line 222]
BUTTON_DONE = done [line 221]
BUTTON_EDIT = edit [line 226]
BUTTON_GO = go [line 225]
BUTTON_NO = no [line 224]
BUTTON_OK = ok [line 218]
BUTTON_SAVE = save [line 220]
BUTTON_YES = yes [line 223]
F_ALPHANUMERIC = alphanumeric [line 200]
```

F_CHECKBOX = checkbox [line 207]
 F_CSRFTOKEN = csrftoken [line 213]
 F_DATE = date [line 203]
 F_DATETIME = datetime [line 205]
 F_FILE = file [line 211]
 F_INTEGER = integer [line 201]
 F_LISTBOX = listbox [line 208]
 F_PASSWORD = password [line 206]
 F_RADIO = radio [line 209]
 F_REAL = real [line 202]
 F_RICHTEXT = richtext [line 212]
 F_SUBMIT = submit [line 210]
 F_TIME = time [line 204]
 string function accesskey_from_string(\$string) [line 1557]

Function Parameters:

- *string* **\$string** the string to process

return the ASCII-character that follows the first tilde in a string

this returns the ASCII-character that follows the first tilde in the string (if any). This is the character that could be added as an accesskey to some HTML tag, e.g. a label or an input. The character is converted to upper case.

Note that a tilde followed by a UTF-8 sequence of 2 or more bytes does NOT yield a hotkey at all but an empty string instead.

string function accesskey_tilde_to_underline(\$string, [\$tag_open = '<u>'], [\$tag_close = '</u>']) [line 1528]

Function Parameters:

- *string* **\$string** the string to process
- *string* **\$tag_open** the tag that starts the emphasis
- *string* **\$tag_close** the tag that ends the emphasis

replace tilde+character with emphasised character to indicate accesskey

this replaces the combination of a tilde and the character following it with \$tag_open followed by the character followed by \$tag_close.

Example: accesskey_tilde_to_underline("~Username") yields
 "<u>U</u>ername"
 accesskey_tilde_to_underline("Pass~word",",") yields "Password"
 accesskey_tilde_to_underline("~Role",",") yields "Role"

Note that we only accept ASCII here: if a tilde is followed by a non-ASCII-character (e.g. the

first byte of a multibyte UTF-8 sequence) we silently ignore and still 'eat' the tilde.

array function dialog_buttondef(\$button_type, [\$value = "], [\$title = "]) [line 1390]

Function Parameters:

- *string* **\$button_type** one of the predefined button constants, e.g. BUTTON_OK
- *string* **\$value** the label used for display including hotkey, eg. '~Yes' or '~Cancel'
- *string* **\$title** the text displayed via a mouseover

shortcut for generating a dialogdef for a button

this constructs an array describing a button. The button definition is bare bones but it includes name, class, value and optionally a title. If no \$value is specified, a translated value is retrieved for the button. If no \$title is specified, a title indicating the hotkey is constructed. This more or less works around the problem that hotkeys cannot be visualised in an input type="submit" button (It is possible in a button tag, but we don't use that because it requires HTML 4. Maybe later...)

If \$button_type is not one of the defined values BUTTON_* the variable itself is used as name, class and as a key for the translation of value unless a \$value is specified.

Typical use: \$dialogdef = array(...,'button_ok' => dialog_buttondef(BUTTON_OK));

Another example defining a non-standard button: \$button = dialog_buttondef('button_foo','~Foo','Do not press this button!');

array function dialog_csrf_token() [line 1469]

shortcut for generating a dialogdef for a csrf_token

this constructs an array describing the csrf_token. Basically we are constructing a hidden field already filled with data from the current session (see [get_csrf_token\(\)](#)). Since get_csrf_token() already fills in the 'name' and 'value', we only have to add the 'type' and set the field to 'hidden'.

string function dialog_get_class(&\$item, [\$class = NULL]) [line 1489]

Function Parameters:

- *array* **&\$item** the parameters that describe the dialog input element
- *string* **\$class** class to start with, otherwise use \$item['class']

construct a space-delimited list of classes that apply to this item

this constructs a string with applicable classes for this element. if the item has validation errors, the class `ATTR_CLASS_ERROR` is added, if the item is viewonly, the class `ATTR_CLASS_VIEWONLY` is added. This allows for the CSS to change the style depending on these situations.

string function `dialog_get_label(&$item)` [line 312]

Function Parameters:

- **array &\$item** the parameters that describe the dialog input element

construct a label for a dialog input element

this constructs the label for an input. It is built inside a label-tag, possibly with these attributes: `id`, `for`, `accesskey`, `class`, `title`. The class is a special case: if there were errors the class `'ATTR_CLASS_ERROR'` is added to the class attribute, in case of viewonly the class `ATTR_CLASS_VIEWONLY` is added too. (Note that the latter shouldn't happen: how can a viewonly field yield any errors at all unless someone is trying to crack the program?)

Because some browsers require that the label of a listbox is linked to the select tag (done via `'id'` and `'for'`), we **MUST** have some `'id'` for that particular tag. If it is not there, we generate one and add it automatically.

- **TODO** if we let the hotkey from the label prevail and add it to the input tag, why add a hotkey to the label too?

array/string function `dialog_get_widget(&$item)` [line 372]

Function Parameters:

- **array &\$item** the parameters that describe the dialog input element

construct an actual HTML input widget for dialog input element

this constructs the actual HTML-code for a dialog input element. If the input element is `'hidden'`, we generate a minimalistic hidden field with no labels, accesskeys or whatever: basically just a name-value-pair to communicate to a subsequent form. Note that we force a

field of type F_CSRFTOKEN to be hidden too, even if it isn't set to be hidden.

If the item is genuine (it has a name and a type), we construct the input using a workhorse routine.

- **TODO** we now only cater for buttons via input type="submit" without the option to visualise the accesskey. Using the button tag could solve that, but button is not defined before HTML 4.01. What to do?
- **TODO** we could manipulate the title attribute of input strings like "please enter a number between {MIN} and {MAX}" based on the various value properties instead of just displaying the title. oh well, for a future version, perhaps...
- **Used by** [Theme::get_jumpmenu\(\)](#)

array/string function dialog_get_widget_file(&\$item, \$name, \$value) [line 1317]

Function Parameters:

- *array* **&\$item** the parameters that describe the dialog input element
- *string* **\$name** the name of the input element ('fieldname')
- *mixed* **\$value** the (current) value of the input element to show ('field value')

construct an input field for file upload

this constructs an input widget for uploading files. This usually includes a button to browse the user's local file system (depends on browser).

Note that it is NOT possible to 'preload' a value in this input field; any predefined value is ignored by the browser. As a workaround we could show the \$value to the user using an additional comment, e.g. by adding it to the label or something. For now, we simply do nothing with the value.

The properties recognised translate to the following HTML-code/attributes

name	: name
value	: value (see note above)
accesskey	: accesskey
columns	: cols (textarea) or size (input type="text")
alt	: alt
class	: class (also depends on viewonly and errors)
tabindex	: tabindex
id	: id
title	: title

multiple : multiple
viewonly : disabled AND addition of ATTR_CLASS_ERROR to class list (if viewonly == TRUE)
errors : addition of ATTR_CLASS_ERROR to class list (if errors > 0)

- **TODO** if we let the hotkey from the label prevail and add it to the input tag, why add a hotkey to the label too?
- **TODO** should we do something with an um-empty \$value? If so, waht? The browser ignores this...

array function dialog_get_widget_listbox(&\$item, \$name, \$value, \$f_type) [line 1044]

Function Parameters:

- *array* **&\$item** the parameters that describe the dialog input element
- *string* **\$name** the name of the input element ('fieldname')
- *mixed* **\$value** the (current) value of the input element to show ('field value')
- *string* **\$f_type** the type of the field (eg text, number, date, time, ...)

construct a listbox

this constructs a listbox

The number of lines in the result depends on the number of items in the options array in \$item. The result always starts with a SELECT opening tag, followed by N OPTION tags and finally a SELECT closing tag.

There are two different ways to specify the options. The simple way is to have a single options array with 'value' => 'option text' pairs. In this case the available properties such as title, class and viewonly are copied from the corresponding generic properties in the \$item array,

The other way is to have an array of arrays like this:

```
$item['options'] = array('1'=>array('title'=>'...', 'option'=>'...'), 2 => array(...));
```

This allows for setting properties of individual options, e.g. one of the options could be made viewonly while the others are still selectable.

The properties recognised translate to the following HTML-code/attributes
name : name
value : value AND perhaps 'selected' if value matches option value
accesskey : accesskey

alt : alt
class : class (also depends on viewonly and errors)
tabindex : tabindex
id : id
title : title
viewonly : disabled AND addition of ATTR_CLASS_ERROR to class list (if viewonly == TRUE)
errors : addition of ATTR_CLASS_ERROR to class list (if errors > 0)
label : tilde+letter may change the accesskey

Note that the options within the SELECT tag are indented 2 spaces for readability.

array/string function dialog_get_widget_radiocheckbox(&\$item, \$name, \$value, \$f_type) [line 919]

Function Parameters:

- *array* **&\$item** the parameters that describe the dialog input element
- *string* **\$name** the name of the input element ('fieldname')
- *mixed* **\$value** the (current) value of the input element to show ('field value')
- *string* **\$f_type** the type of the field (eg text, number, date, time, ...)

construct a checkbox or 1 or more radiobuttons

this constructs a checkbox or a list of radiobuttons.

Note: because a checkbox and radionbuttons are very similar, they are handled in the same workhorse routine. Maybe we should split this in the name of code clarity. Oh well...

If we are generating a checkbox, the result looks something like this: <input type="checkbox" value="1" checked ...><label ...>option text</label>

If we are generating radiobuttons, the result looks something like this: <input type="radio" value="1" checked ...><label ...>option 1 text</label>

<input type="radio" value="2" ...><label ...>option 2 text</label>

<input type="radio" value="3" ...><label ...>option 3 text</label>

The number of lines in the result depends on the number of items in the options array in \$item. In case of a checkbox there should only be one, in case of radiobuttons there should be more than 1.

There are two different ways to specify the options. The simple way is to have a single options array with 'value' => 'option text' pairs. In this case the available properties such as title, class and viewonly are copied from the corresponding properties in the \$item array,

The other way is to have an array of arrays like this:

```
$item['options'] = array('1'=>array('title'=>'...', 'option'=>'...'), 2 => array(...));
```

This allows for setting properties of individual options, e.g. one of the radio buttons could be made viewonly while the others are still selectable.

Note that such a non-simple array of arrays doesn't make much sense for a single, simple checkbox.

The properties recognised translate to the following HTML-code/attributes

name	: name
value	: value AND perhaps 'checked' if value matches option value
accesskey	: accesskey
alt	: alt
class	: class (also depends on viewonly and errors)
tabindex	: tabindex
id	: id
title	: title
viewonly	: disabled AND addition of ATTR_CLASS_ERROR to class list (if viewonly == TRUE)
errors	: addition of ATTR_CLASS_ERROR to class list (if errors > 0)
label	: tilde+letter may change the accesskey

Note: In case of radiobuttons, the document-wide unique id is constructed from the specified id by appending an underscore and an indexnumber (except for the first item in the list). This id can be overruled by an id specified in the options array-of-arrays.

Even when an item has an explicit accesskey, the access key can be overruled by the 'hotkey' derived from a tilde+letter combination in the options array, either in the simple case or in case of an array-of-arrays.

Note that also the generic title can be overruled by a title that is defined in the options array-of-arrays.

array/string function `dialog_get_widget_richtextinput(&$item, $name, $value, $f_type)` [line 1149]

Function Parameters:

- *array* **&\$item** the parameters that describe the dialog input element
- *string* **\$name** the name of the input element ('fieldname')
- *mixed* **\$value** the (current) value of the input element to show ('field value')
- **\$f_type**

construct an input field using the user's preferred editor

this constructs an input for rich text like a page with HTML-code. Most users will probably have selected the CKeditor. However, there is also the option to use the so-called 'plain' editor, which is nothing more than a textarea in disguise.

The properties recognised translate to the following HTML-code/attributes

name	: name
value	: value
accesskey	: accesskey
rows	: rows
columns	: cols
maxlength	: maxlength
alt	: alt
class	: class (also depends on viewonly and errors)

tabindex : tabindex
id : id
title : title
viewonly : disabled AND addition of ATTR_CLASS_ERROR to class list (if viewonly == TRUE)
errors : addition of ATTR_CLASS_ERROR to class list (if errors > 0)

For maximum compatibility we not only retain FCKEditor 2.6.5 (2359) but also CKEditor 3.6.6.2 (7696) next to CKEditor 4.4.5 (25cdcad). The actual editor to use depends on the system wide choice and user preference. For compatibility the latest editor is configured to be 'ckeditor' whereas the older version 3 is now selected with 'ckeditor3'. This means that existing 'ckeditor'-users will be moved to CKEditor4 whereas 'plain' and 'fckeditor' users will see no change. (November 2014).

- **TODO** if we let the hotkey from the label prevail and add it to the input tag, why add a hotkey to the label too?

array/string function dialog_get_widget_submit(&\$item, \$name, \$value, \$f_type) [line 810]

Function Parameters:

- *array* **&\$item** the parameters that describe the dialog input element
- *string* **\$name** the name of the input element ('fieldname')
- *mixed* **\$value** the button's label possibly including a tilde indicating hotkey
- *string* **\$f_type** the type of the field (eg text, number, date, time, ...)

construct a submit button

this constructs a submit button. For compatibility we use a simple input of type submit because the button widget is only available since HTML 4. We may change that in the future, and force everyone to use at least HTML 4. For now it is as it is.

Note that the label of the button is retrieved from \$value rather than from the label property. We do use the \$value as a string possibly containing hotkeys (via prepending a letter with a tilde) and we also set the accesskey to that value. However, it is different from other widgets because an input cannot display underlines (a button can).

The properties recognised translate to the following HTML-code/attributes
name : name
value : value
accesskey : accesskey

alt : alt
 class : class (also depends on viewonly and errors)
 tabindex : tabindex
 id : id
 title : title
 viewonly : disabled AND addition of ATTR_CLASS_ERROR to class list (if viewonly == TRUE)
 errors : addition of ATTR_CLASS_ERROR to class list (if errors > 0)

array/string function `dialog_get_widget_textinput(&$item, $name, $value, $f_type)` [line 716]

Function Parameters:

- *array* **&\$item** the parameters that describe the dialog input element
- *string* **\$name** the name of the input element ('fieldname')
- *mixed* **\$value** the (current) value of the input element to show ('field value')
- *string* **\$f_type** the type of the field (eg text, number, date, time, ...)

construct an input field, usually for text input OR a textarea for multiline input

this constructs most variations on text fields, including password fields. Many of the defined field types (the F_* constants) can be handled via a simple input of type text. The semantics of the field (eg. is it an integer, a real) have no impact on the HTML-input: at that level it is still plain text. However, for a password we use the password type in order to make the value display as asterisks. If the number of rows is more than 1, the input element becomes a text area. Note that this generally only applies to F_ALPHANUMERIC but that is not enforced here (you can make a multiline F_DATE, even though it doesn't make much sense).

The properties recognised translate to the following HTML-code/attributes

name : name
 value : value
 accesskey : accesskey
 rows : rows (textarea only)
 columns : cols (textarea) or size (input type="text")
 maxlength : maxlength
 alt : alt
 class : class (also depends on viewonly and errors)
 tabindex : tabindex
 id : id
 title : title
 autocomplete : autocomplete [1]
 viewonly : disabled AND addition of ATTR_CLASS_ERROR to class list (if viewonly == TRUE)
 errors : addition of ATTR_CLASS_ERROR to class list (if errors > 0)

[1] Note that autocomplete is a non-standard function that doesn't work in every browser.

- **TODO** if we let the hotkey from the label prevail and add it to the input tag, why add a hotkey to the label too?

array function dialog_quickform(\$href, &\$dialogdef, [\$method = 'post'], [\$attributes = ""]) [*line 250*]

Function Parameters:

- *string* **\$href** the target of the HTML form
- *array* **&\$dialogdef** the array which describes the complete dialog
- *string* **\$method** method to submit data to the server, either 'post' or 'get'
- *string/array* **\$attributes** holds the attributes to add to the form tag

construct a generic form with a dialog

this constructs an HTML form with a simple dialog where

- every label and every widget has its own line
(enforced by a BR-tag)
- label/widget-combinations are separated with a P-tag
- buttons are stringed together on a single line (ie no trailing BR)

This should be sufficient for many dialogs. If the layout needs to be more complex a custom dialog can always be constructed using functions [dialog_get_label\(\)](#) and [dialog_get_widget\(\)](#).

- **Usedby** [ConfigAssistant::show_dialog\(\)](#)
- **Uses** [html_form\(\)](#)

bool function dialog_validate(&\$dialogdef) [*line 456*]

Function Parameters:

- **array `&$dialogdef`** the complete dialog definition; contains detailed errors and/or reformatted values

validate and check values that were submitted via a user dialog

this steps through the definition of a dialog and retrieves the values submitted by the user via `$_POST[]`. The values are checked against the constraints (e.g. minimum string length, date range, etc.). If the submitted value is considered valid, it is stored in the corresponding value of the `dialogdef` element, maybe properly reformatted (in case of dates/times/datetimes). If there were errors, these are recorded in the dialog definition element, in the form of one or more readable error messages. Also the error count (per element) is incremented. This makes it easy to

- inform the user about what was wrong with the input data
- determine whether there was an error at all (if `$dialogdef[$k]['errors'] > 0`).

Note that this routine has the side effect of filling the dialog array with the data that was submitted by the user via `$_POST`. If the validation is successful, the data is ready to be saved into the database. If it is not, the data entered is still available in the `dialogdef` which makes it easy to return to the user and let the user correct the errors without losing all the data input because of a silly mistake in some input field.

Update 2009-03-17: We no longer validate the view-only fields because these fields are not POST'ed by the browser and hence cannot be validated. This also means that there is no value set from `$_POST` for those fields.

Update 2011-09-29: added UTF-8 validation, replace with U+FFFD (Unicode replacement character) on fail

- **TODO** add an error message to
- **Usedby** [ConfigAssistant::save_data\(\)](#)

bool function valid_datetime(\$f_type, \$input, &\$output) [line 1593]

Function Parameters:

- **string `$f_type`** indicates the field type we are expecting, can be `F_DATE`, `F_TIME` or `F_DATETIME`
- **string `$input`** the string that needs to be checked
- **string `&$output`** if the input is valid, this contains a properly formatted value

check validity of date, time or datetime

this checks the validity of dates and times. If all tests are passed successfully, the input value is reformatted in the standard format corresponding with that field type:

- F_DATE becomes yyyy-mm-dd (with leading zeros for month or day where applicable)
- F_TIME becomes hh:mm:ss (with leading zeros when applicable)
- F_DATETIME is combination of F_DATE and F_TIME glued together with a space: yyyy-mm-dd hh:mm:ss

Valid values for dates are within the range 0000-01-01 ... 9999-12-31 (but note that the year is always displayed with 4 digits). This routine takes leap years into account the same way the standard function `checkdate()` does.

Valid values for times are between 00:00:00 and 23:59:59. Note that we don't deal with leap seconds or other fancy stuff (this is not rocket science): KISS. Usually we only need times to determine an embargo date/time anyway.

Also, this routine doesn't know about time zones and daylight savings time.

donors.php

/program/lib/donors/donors.php - a list of benefactors (people and organisations)

This file provides a list of people and organisations that contributed to Website@School by donating money, commissioning specific features, etc. It is included and called from /program/main_admin.php.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: donors.php,v 1.8 2016/06/14 09:28:15 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

void function show_benefactor_logo(&\$output, [\$text_only = FALSE], [\$num = 1], [\$m = ""]) [*line 39*]

Function Parameters:

- *object* **&\$output** collects the html output
- *bool* **\$text_only** if TRUE do not show a graphical image
- *int* **\$num** the number of benefactors to show this time around
- *string* **\$m** margin for increased readability

output the logos of zero, one or more of the Website

email.class.php

/program/lib/email.class.php - wrapper for sending mail

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: email.class.php,v 1.11 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

filelib.php

/program/lib/filelib.php - utilities for manipulating files

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: filelib.php,v 1.11 2016/06/03 13:44:46 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool|string function file_available(\$filename) [*line 168*]

Function Parameters:

- *string* **\$filename** is the requested file

check availability of requested file for the current user

This checks the \$filename for permissions and existence. Returns FALSE if the current user is not allowed to see the file or if the file does not exist. We remember failures in the logger.

bool|string function get_mediatype(\$mimetype) [*line 148*]

Function Parameters:

- *string* **\$mimetype** the full mimetype to examine, possibly with parameters

extract the mediatype and -subtype from a full mimetype

this extracts the mediatype and -subtype from a full mimetype, i.e. 'text/plain' from 'text/plain; charset=US-ASCII' (see also [get_mimetype\(\)](#) and RFC2616). If \$mimetype doesn't look like a mimetype, we return FALSE.

- Usedby [send_file_from_datadir\(\)](#)
- Usedby [get_mimetype\(\)](#)

string function `get_mimetype($path, [$name = ''])` [line 82]

Function Parameters:

- *string* **\$path** fully qualified path to the file to test
- *string* **\$name** name of the file, possibly different from \$path

determine the mimetype of a file

Try to discover the mimetype of a file. First we see if `getimagesize()` yields results; if so we're done (0). Very efficient if there are many image files and not so many other files.

If not an image we try `finfo-method` (1), `mime_content_type()`-method (2) and finally the `shell-to-file(1)`-method (3) to determine the mime type.

Note that in step 3 we shell out and try to execute the `file(1)` command. The results are checked against a pattern to assert that we are really dealing with a mime type. The pattern is described in RFC2616 (see sections 3.7 and 2.2) (see also [get_mediatype\(\)](#)):

```
media-type  = type "/" subtype *( ";" parameter )
type        = token
subtype     = token
token       = 1*<any CHAR except CTLs or separators>
separators  = "(" | ")" | "<" | ">" | "@"
             | "," | ";" | ":" | "\" | "<">
             | "/" | "[" | "]" | "?" | "="
             | "{" | "}" | SP | HT
CHAR        = <any US-ASCII character (octets 0 - 127)>
CTL         = <any US-ASCII control character
             (octets 0 - 31) and DEL (127)>
SP          = <US-ASCII SP, space (32)>
HT          = <US-ASCII HT, horizontal-tab (9)>
"<">       = <US-ASCII double-quote mark (34)>
```

This description means we should look for two tokens containing letters a-z or A-Z, digits 0-9 and these special characters: `! # $ % & ' * + - . ^ _ ` |` or `~`. That's it.

Note that some methods may return a mime type with additional parameters. e.g. `'text/plain; charset=US-ASCII'`. This fits the pattern, because it starts with a token, a slash and another token.

The optional parameter `$name` is used to perform special handling of the text/css type: some methods (notably old versions of `file(1)`) return 'text/plain' instead of 'text/css'. Workaround: if we see a mimetype starting with 'text/' AND the extension is .css we assume text/css.

Note that we trust the user at least partially(`$name` eventually originales in the `$_FILES[]` array). The path to the actual file to check is in `$path`. If `$name` is not specified, we extract the extension from `$path` (as in the case of [send_file_from_datadir\(\)](#)).

- Usedby [send_file_from_datadir\(\)](#)
- Uses [get_mediatype\(\)](#)

bool|string function `mime_ext_by_typ($typ)` [line 279]

Function Parameters:

- *string* **\$typ** mime type

lookup extension in table, using mimetype as key

return the first extension associated with mimetype `$typ`

- Uses `$MIME_TYP_EXT`

bool|string function `mime_typ_by_ext($ext)` [line 298]

Function Parameters:

- *string* **\$ext** filename extension

lookup mimetype in table, using extension as key

return the first mimetype associated with filename extension `$ext`

- **Uses** \$MIME_EXT_TYP

bool function mime_typ_ext_match(\$typ, \$ext) [*line 256*]

Function Parameters:

- *string* **\$typ** mimetype
- *string* **\$ext** filename extension

verify the combination of filename extension and mimetype

this looks up the mimetypes associated with filename extension \$ext in the global table \$MIME_EXT_TYP. If any of those mimetypes matches the requested mimetype we confirm that ext and typ match.

Note 1: There is another table: \$MIME_TYP_EXT that works the other way around. We could have used either table; I choose the former.

Note 2: The two tables are generated with a development tool (see /devel/tools) . This tool reads the existing table in this file filelib.php and optionally a newer version of the Apache mime.types file creating updated tables.

- **Uses** \$MIME_EXT_TYP

filemanager.class.php

/program/lib/filemanager.class.php - filemanager

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: filemanager.class.php,v 1.23 2016/06/02 16:28:12 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

PARAM_FILENAME = filename_ [line 41]

This constant is used to construct the fieldname used for deleting files

PARAM_FILENAMES = filenames [line 44]

This constant is used to construct the fieldname counting the number of files to delete

PARAM_PATH = path [line 46]
PARAM_SORT = sort [line 47]
SORTBY_DATE_ASC = 3 [line 53]
SORTBY_DATE_DESC = -3 [line 54]
SORTBY_FILE_ASC = 1 [line 49]
SORTBY_FILE_DESC = -1 [line 50]
SORTBY_NONE = 0 [line 48]
SORTBY_SIZE_ASC = 2 [line 51]
SORTBY_SIZE_DESC = -2 [line 52]
TASK_ADD_DIRECTORY = mkdir [line 37]
TASK_ADD_FILE = upload [line 36]
TASK_CHANGE_DIRECTORY = cd [line 32]
TASK_LIST_DIRECTORY = ls [line 31]
TASK_PREVIEW_FILE = preview [line 33]
TASK_REMOVE_DIRECTORY = rmdir [line 35]
TASK_REMOVE_FILE = rm [line 34]
TASK_REMOVE_MULTIPLE_FILES = batchrm [line 38]
require_once **\$CFG->progdire**."/lib/filelib.php" [line 28]

utility routines for manipulating files

groupmanager.class.php

/program/lib/groupmanager.class.php - taking care of group management

This file defines a class for dealing with groups.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: groupmanager.class.php,v 1.18 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

GROUPMANAGER_MAX_CAPACITIES = 8 *[line 30]*

this defines the maximum number of capacities a group can have (keep this below 10 because of dialog hotkeys)

htmllib.php

/program/lib/htmllib.php - useful functions for generating HTML-code

This file provides various utility routines that aid in creating HTML-code.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: htmllib.php,v 1.10 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

void function href(\$path, [\$params = "], [\$fragment = "]) [line 222]

Function Parameters:

- *string* **\$path** the hypertext reference
- *array|string* **\$params** the parameter(s) to add to the \$path
- *string* **\$fragment** the optional position within the page

construct a href from a path, params and a fragment

- **TODO** should we merge this with `html_a()` and/or rename this routine to `html_href()`?

string function html_a([\$href = "], [\$params = NULL], [\$attributes = NULL], [\$anchor = NULL], [\$fragment = NULL]) [line 74]

Function Parameters:

- *string* **\$href** holds the hypertext reference
- *string|array* **\$params** holds the parameters to add to the \$href

- *string/array* **\$attributes** holds the attributes to add to the tag
- *string* **\$anchor** if not empty this string and a closing tag are appended
- *string* **\$fragment** the optional position within the targeted page

construct an HTML A tag with optional parameters and attributes and optional fragment

this constructs an A tag of the form ``

If no parameters are specified, nothing is added to the href. If no attributes are specified the tag only has the href attribute. If \$params is an array, the elements of this array are added to the href after a `rawurlencode()`. The complete \$href is then escaped via `htmlspecialchars()`. If \$attributes is an array, all elements are added as escaped key-value-pairs. If string, then just append. If \$anchor is not empty, the string is appended to the constructed opening tag and subsequently a closing tag is appended.

Note that urlencoding andspecialchars are applied to the URL property and that the other properties are only are htmlspecialchars()'ed. The optional \$anchor is not changed in any way.

Examples: `html_a('index.php')`: ``
`html_a('index.php',array('foo'=>'bar'))`: ``
`html_a('index.php',array('x'=>'y'),array('title'=>'foo'))`: ``
`html_a('index.php',"",array('class'=>'dimmed'),'baz')`: `baz`
`html_a("","",array('name'=>'chapter1'),'chapter 1')`: `chapter 1`

Note: somehow the parameter \$fragment never made it to earlier versions of `html_a()`. This parameter was added at the end of the parameter list for compatibility reasons (2014-05-13/PF)

string function `html_attributes($attributes)` [line 177]

Function Parameters:

- *mixed* **\$attributes** an array or string with attributes or NULL

convert an array of name-value pairs to a string

this converts an array of name-value-pairs to a string containing attribute="content" items, where both 'attribute' and 'content' are properly escaped (with `htmlspecialchars()`). Properties that don't have content, such as 'disabled' or 'selected' or 'checked' can be specified using the special value NULL, e.g. `array('disabled' => NULL)`. If the parameter \$attributes happens to be a string, it is returned with a space

prepended. If it is neither a string nor an array (e.g. NULL), an empty string is returned. Note that the attribute="content" elements are delimited with spaces, and that a leading space is prepended (but not trailing space is added).

string function `html_form($action, [$method = 'post'], [$attributes = ''])` [line 200]

Function Parameters:

- *string* **\$action** the url to submit to
- *string* **\$method** either get or post (default)
- *string|array* **\$attributes** holds the attributes to add to the tag

construct the opening of a HTML form

- Used by [dialog_quickform\(\)](#)
- Used by [TranslateTool::render_translation_dialog\(\)](#)

void function `html_form_close()` [line 209]

companion of `html_form`: close the tag

string function `html_img([$src = ''], [$attributes = NULL])` [line 123]

Function Parameters:

- *string* **\$src** holds the url to the image file
- *string|array* **\$attributes** holds the attributes to add to the tag

construct an HTML IMG tag with optional attributes

this constructs an IMG tag of the form ``

If no attributes are specified the tag only has the src attribute. If \$attributes is an array, all elements are added as raw encoded key-value-pairs. If it is a string, then just append.

Examples: `html_img('icon.gif')`: ``

`html_img('icon.gif', array('width'=>16, 'height'=>16))`: `

void function html_input_radio(\$name, \$options) [line 259]

Function Parameters:

- **\$name**
- **\$options**

STUB

void function html_input_select(\$name, \$options) [line 249]

Function Parameters:

- **\$name**
- **\$options**

STUB

void function html_input_submit(\$name, \$value) [line 268]

Function Parameters:

- **\$name**
- **\$value**

STUB

void function html_input_text(\$name) [line 243]

Function Parameters:

- **\$name**

STUB

string function `html_table`([\$attributes = NULL], [\$content = NULL], \$m) [*line 279*]

Function Parameters:

- *string/array* **\$attributes** holds the attributes to add to the tag
- *string* **\$m** margin for improved code readability
- **\$content**

construct the opening of a HTML table

void function `html_table_cell`([\$attributes = NULL], [\$content = NULL]) [*line 304*]

Function Parameters:

- **\$attributes**
- **\$content**

void function `html_table_cell_close`() [*line 309*]

string function `html_table_close`([\$m = margin for improved code readability]) [*line 289*]

Function Parameters:

- *string* **\$m** margin for improved code readability

construct table closing tag

void function `html_table_head`([\$attributes = NULL], [\$content = NULL]) [*line 314*]

Function Parameters:

- **\$attributes**
- **\$content**

void function `html_table_head_close`() [*line 319*]

void function `html_table_row`([\$attributes = NULL], [\$content = NULL]) [*line 294*]

Function Parameters:

- **\$attributes**

- **\$content**

void function `html_table_row_close()` [*line 299*]

string function `html_tag($tag = "", [$attributes = NULL], [$content = NULL])` [*line 141*]

Function Parameters:

- *string* **\$tag** is the HTML-tag to create, e.g. 'span' or 'script'
- *mixed* **\$attributes** holds the attributes to add to the tag or NULL
- *mixed* **\$content** if not NULL this string and a closing tag are appended

construct a generic HTML-tag with attributes, optionally close it too

string function `html_tag_close($tag = "")` [*line 155*]

Function Parameters:

- *string* **\$tag** is the HTML-tag to close, e.g. 'span' or 'script'

companion of `html_tag`: close the tag

language.class.php

/program/lib/language.class.php - taking care of translations of messages

This file defines a class for dealing with translation of phrases.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: language.class.php,v 1.6 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/lib/loginlib.php -- functions to handle user login/logout

Visitors need to authenticate when they want to see a 'protected' area or when they want to modify the website content. This requires a user account and the visitor presenting valid credentials (username + password).

We don't want malicious scripts trying to get in with brute force. However, we need to accomodate users that make typo's while entering credentials. Also we want to allow for sending password reminders, in a safe way.

Features:

- users are allowed N login attempts within an interval of T1 minutes
 - users can request a new password (a 'bypass') to be mailed to them.
this additional password is valid for only T2 minutes
 - if a user has requested a bypass, the user is forced to change her password. the new password must differ from the old password and also from the bypass
 - if too many failures are detected in the last T1 minutes, login attempts from the corresponding IP-address are blocked for T3 minutes
- N = \$CFG->login_max_failures, default 10

T1 = \$CFG->login_failures_interval, default 12 minutes

T2 = \$CFG->login_bypass_interval, default 30 minutes

T3 = \$CFG->login_blacklist_interval, default 8 minutes

Once a user is authenticated, a PHP-session is established, using our own database based session handler. The session key is stored in a cookie in the user's browser. Presenting this cookie on subsequent calls is enough to gain access. The logout routine takes care of killing both the user's cookie and the session in the database.

There are several different login procedures.

1. Normal login

The user enters a valid username and password and is subsequently logged in.

2. Change password The user enters a valid username and password and also a valid new password (twice). A salted hash of the new password is recorded in the database and the user is logged in.

3. Forgotten password, phase 1: sending a laissez-passer The user presents a valid combination of username and email address. Subsequently a one-time logon-code (dubbed 'laissez-passer') is sent to the user's email address. This code is valid for at most T2 minutes. This code can be used, exactly once, to send a temporary password via email.

4. Forgotten password, phase 2: sending a temporary password The user clicks the link

received in phase 1 and a temporary password (dubbed 'bypass') is sent to the user. This temporary password is valid for another T2 minutes.

5. Message box This is a pseudo-procedure. A simple 'message box' type of screen is displayed but no real interaction is anticipated via this screen. This is used to tell the user that things didn't work out (too many failures) or to check their mail for further instructions (e.g. when a laissez passer was sent). Whenever the user acknowledges this screen by clicking the button, she usually is directed to \$WAS_SCRIPT_NAME.

6. Blacklist This is also a pseudo-procedure. The corresponding number is used to identify blacklisted IP-addresses in the database.

Note that when the user logs in after a temporary password has been sent, the normal login procedure is immediately followed by a (forced) 'change password' procedure. This makes sure that a temporary password will be changed immediately after the user logs in.

Note that each of the procedures can be entered 'manually', i.e. by opening `index.php?login=X` the user starts procedure X. This allows for the user to change her password whenever she feels this is necessary, without going through the trouble of the 'forgotten password'-procedure which eventually ends with the user changing her password too.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: loginlib.php,v 1.19 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **TODO** should we suppress the username in the laissez-passer routine? We _do_ leak the the username in an insecure email message. This does require making the laissez-passer code unique in the database (currently only username+code has to be unique and that's easy because the username itself is unique).
- **TODO** should we normalize the remote_addr everywhere? We now rely on the remote_addr being equal to some stored value (in the database) but with an IPv6 address there are several possibilities to have different representations of the same address (e.g. '::dead:beef' is equivalent to '::0:dead:beef' or even ' '::DeAd:BeeF' or '0000:0000:0000:0000:0000:0000:DEAD:BEEF'. This problem also exists with IPv4: '127.0.0.1' is equivalent to '127.000.000.001'. *sigh*
- **Usedby** [mypage_view_password\(\)](#)
- **Usedby** [admin_show_login_and_exit\(\)](#)
- **Usedby** [admin_login\(\)](#)
- **Usedby** [admin_logout_and_exit\(\)](#)

- License [GNU AGPLv3+Additional Terms](#)

BY_EMAIL = 2 *[line 156]*

this selects authentication via username+email in authenticate_user()

BY_LAISSEZ_PASSER = 3 *[line 159]*

this selects authentication via username+laissez_passer in authenticate_user()

BY_PASSWORD = 1 *[line 153]*

this selects authentication via username+password in authenticate_user()

LOGIN_FAILURE_DELAY_SECONDS = 3 *[line 138]*

this is the number of seconds to delay responding after a login action fails (slow 'm down..)

LOGIN_PROCEDURE_BLACKLIST = 6 *[line 135]*

this is a pseudo procedure, used to record blacklisted IP-addresses

LOGIN_PROCEDURE_CHANGE_PASSWORD = 2 *[line 123]*

this is the procedure to change the user's password

LOGIN_PROCEDURE_MESSAGE_BOX = 5 *[line 132]*

this is a pseudo procedure, used to deliver some message to the user

LOGIN_PROCEDURE_NORMAL = 1 *[line 120]*

this is the usual procedure for logging in

LOGIN_PROCEDURE_SEND_BYPASS = 4 *[line 129]*

this is phase 2 of the 'forgot password' procedure

LOGIN_PROCEDURE_SEND_LAISSEZ_PASSER = 3 *[line 126]*

this is phase 1 of the 'forgot password' procedure

LOGIN_PROCEDURE_SHOWLOGIN = 0 *[line 117]*

this only shows the login dialog

MINIMUM_PASSWORD_DIGITS = 1 *[line 150]*

this is the hardcoded minimal number of digits in a new password

`MINIMUM_PASSWORD_LENGTH = 6 [line 141]`

this hardcoded minimal length is enforced whenever a user wants to change her password

`MINIMUM_PASSWORD_LOWERCASE = 1 [line 144]`

this is the hardcoded minimal number of lower case characters in a new password

`MINIMUM_PASSWORD_UPPERCASE = 1 [line 147]`

this is the hardcoded minimal number of upper case characters in a new password

`bool function acceptable_new_password($new_pass1, $new_pass2, [$salt = ''], [$password_hash = ''],
[$bypass_hash = '']) [line 1090]`

Function Parameters:

- *string* **\$new_pass1** new password
- *string* **\$new_pass2** new password again, to prevent typo's
- *string* **\$salt** (optional) the salt that was used to hash the old password and the bypass
- *string* **\$password_hash** (optional) the hashed existing password
- *string* **\$bypass_hash** (optional) the hashed bypass password

check the new passwords satisfy password requirements

Users should provide the same password twice, to prevent typo's, so both passwords should be equal. Also, the following requirements should be satisfied:

- the minimum password length should be `MINIMUM_PASSWORD_LENGTH` (default 6)
- the new password should contain at least `MINIMUM_PASSWORD_LOWERCASE` lowercase letter a-z (default 1)
- the new password should contain at least `MINIMUM_PASSWORD_UPPERCASE` upper case letter A-Z (default 1)
- the new password should contain at least `MINIMUM_PASSWORD_DIGITS` digit 0-9 (default 1)
- the new password should not be the same as the previous password
- the new password should not be the same as the bypass password (if any was issued)

Note that the bypass-generator also should satisfy these rules. This could lead to the thought of accepting the bypass-password as the permanent one. However, since this temporary password was sent to the user in a plain-text email message, we should consider this a 'bad' password.

The minimum password length and other minimum values are not configurable (via `$CFG`)

because that would make it too easy (too tempting) to give in and use weak passwords (too short, only lowercase, etc.) However, if you really **MUST**, you could change the `MINIMUM_PASSWORD_*` constants defined above.

Note that the check againsts existing (temporary and regular) passwords is not performed if the corresponding parameters are empty. If they are empty, this routine only performs the first 4 checks in the list above.

- **TODO** Entropy! Perhaps it is better to value the strength of a password in a different way, say 'CorrectHorseBatteryStaple' could be better than 'Tr0ub4dor&3'. Also, why force a limited set of ASCII-characters [0-9A-Za-z] when there is so much entropy to gain by using characters in the range U+0080 - U+10FFFF? (20140912/PF)
- **Usedby** [mypage_authorisation\(\)](#)
- **Usedby** [mypage_password_dialog_validate\(\)](#)

bool/array function `authenticate_user($by_what_token, $username, $token)` [line 982]

Function Parameters:

- *int* **\$by_what_token** which authentication token to use
- *string* **\$username** username the user entered in the dialog
- *string* **\$token** the token is either password, email or laissez_passer entered by the user

check the user's credentials in one of three ways

This authenticates the user's credentials. There are some variants:

- by password: the user's password should match
 - by email: the user's email should match
 - by laissez passer: the one-time authentication code should match
- Strategy: we first read the active record for user `$username` in core. If there is none, the user does not exist or is inactive => return FALSE.

After that we check the validity of the token:

- a password is checked via the password hash or, if that fails, via the bypass hash. In the latter case, the bypass should not yet be expired (a bypass and a laissez_passer are valid until the 'bypass_expiry' time).
- an email address is checked caseInsensitive and without leading/trailing spaces

- a `laissez_passer` is checked much the same way as the bypass password, but it that the code is stored 'as-is' rather than as a hash. The comparison is caseInsensitive.
- If the credentials are considered valid, an array with the user record is returned, otherwise FALSE is returned.

Because there are actually several checks to be done, we decided not to use SQL like: `SELECT * FROM users WHERE username=$username AND password=$password`, not the least because we need to have the salt in our hands before we can successfully compare password hashes.

Note: The 'special cases' (checking email, checking `laissez_passer`, checking bypass) all have their token stripped from leading and trailing spaces. We don't want to further confuse the user by not accepting a spurious space that was entered in the heat of the moment when the user has 'lost' her password. Therefore we also always trim the username. Rationale: usernames and also the generated passwords etc. never have leading/trailing spaces. However, one cannot be sure that a user has not entered a real password with leading/trailing space, so we do NOT trim the \$token in the first attempt in the case 'BY_PASSWORD' below.

- Used by [mypage_authorisation\(\)](#)

bool function login_change_password(\$user_id, \$new_password) [line 1034]

Function Parameters:

- *int* **\$user_id** identify the user record by user_id
- *string* **\$new_password** the new password in plain text

update the users database with a new (randomly salted) password and reset bypass mode to normal

This updates the user record for user with user_id and stores the new password. The new password and a new random salt are hashed together and the result is stored, together with the new salt, overwriting the old salt and the old password hash. The bypass mode is reset to normal and the bypass hash is reset. Return TRUE on success.

- Used by [mypage_view_password\(\)](#)

array function login_dialogdef(\$dialog) [line 1380]

Function Parameters:

- *int* **\$dialog** indicates which dialog to construct

construct a standard dialog definition for a specific login procedure

There are 5 different dialogs, each with 1, 3 or 5 inputs, depending on \$dialog Field

Used in dialogs:

login_username	1 2 3 4
login_password	1 2
login_new_password1	2
login_new_password2	2
login_email	3
login_laissez_passer	4
button	1 2 3 4 5

Note: There is some discussion about the autocomplete flag. Before HTML5 it was a browser-specific feature that sometimes didn't work. However, we can at least try to prevent it by adding autocomplete="off" to password fields.

bool/int function login_failure_blacklist_address(\$remote_addr, \$delay_in_seconds, [\$username = ""]) [line 1256]

Function Parameters:

- *string* **\$remote_addr** the remote IP-address is the origin of the failure
- *int* **\$delay_in_seconds** the number of seconds to put this address on the blacklist
- *string* **\$username** extra information, could be useful for troubleshooting afterwards

add remote_addr to the blacklist for specified interval (in seconds)

- Used by [mypage_authorisation\(\)](#)

bool/int function login_failure_delay(\$remote_addr) [line 1291]

Function Parameters:

- *string* **\$remote_addr** the remote IP-address that is the origin of the failure

delay execution of this script for a few seconds and blacklist the remote_addr during the delay

This immediately blacklists the remote address for LOGIN_FAILURE_DELAY_SECONDS seconds. Once that is done, the execution is delayed for that same period of time. After the delay, the temporary blacklisting is removed from the table. The whole purpose of this rapid succession of an INSERT and a DELETE is to prevent brute force attack scripts that do not wait for an answer and/or use multiple connections. This routine defeats that trick, because nothing can be done when an IP-address is blacklisted.

- **Usedby** [mypage_authorisation\(\)](#)
- **Uses** \$CFG

int function login_failure_increment(\$remote_addr, \$procedure, [\$username = ""]) [line 1313]

Function Parameters:

- *string* **\$remote_addr** the remote IP-address that is the origin of the failure
- *int* **\$procedure** indicates in which procedure the user failed
- *string* **\$username** extra information, could be useful for troubleshooting afterwards

add 1 point to score for a particular IP-address and failed procedure, return the new score

This records a login failure in a table and returns the the number of failures for the specified procedure in the past T1 minutes.

bool function login_failure_reset(\$remote_addr) [line 1244]

Function Parameters:

- *string* **\$remote_addr** the remote IP-address is the origin of the failure

deactivate all login failures/blacklisting scores for remote_addr

This resets all the scores for all failed login attempts and blacklistings for the specified IP-address. The records in the login_failures table are deactivated by deleting the records for this remote_addr.

Note that the failed logins and the blacklistings are recorded in the log_messages table via [logger\(\)](#). Therefore we can automatically keep this table 'login_failures' clean without cron jobs.

This routine resets `_all_` scores, including any blacklisting that might still be active, i.e. which has a datim in the future.

- Usedby [mypage_authorisation\(\)](#)

bool function login_is_blacklisted(\$remote_addr) [*line 1212*]

Function Parameters:

- *string* **\$remote_addr** the remote IP-address to be checked

find out if a remote address is blacklisted at this time

This routine checks if this remote address is blacklisted in the login_failures table with a datim that lies in the future. If this is the case, the address is indeed blacklisted and TRUE is returned. Note that we sum the points much the same way as in [login_failure_increment](#) rather than counting 'blacklist-records'.

- **TODO** Should we first change \$remote_addr to canonical form for good comparison? (20140912/PF)
- Usedby [mypage_authorisation\(\)](#)

string function login_propagate_navigation(\$path, [\$add = array()], [\$propagate = TRUE]) [*line 1502*]

Function Parameters:

- *string* **\$path** to the PHP-script
- *array* **\$add** has additional parameters to add to the \$action
- *bool* **\$propagate** if FALSE don't propagate after all

construct an action attribute propagating existing parameters

this routine constructs an action value of a form keeping any existing information from \$_SERVER['PATH_INFO'] and/or \$_GET[], perhaps with additional parameters from the \$add-array.

The object of this exercise is to allow the distribution of links like <http://exemplum.eu/index.php/area/2/Intranet.html> that actually work by first letting the user login (via show_login() etc) and subsequently falling through to the correct page in a single pass. This requires that we keep all parameters the user presented initially, including the PATH_INFO and the \$_GET-parameters. However, this routine also adds the parameters in \$add to the mix, allowing the user to return to the login-routine when she is posting her credentials. In other words: we attempt to propagate the place the user wants to navigate to after logging in to the next screen.

Note that it would be weird to propagate the parameter 'logout' while attempting to login; it yields an endless loop and it becomes impossible to login from the screen that is presented. Therefore we unconditionally get rid of the 'logout' parameter while copying \$_GET[]. The same logic applies to the login parameter: if we need it to return to loginlib, it should be set (again) via \$add; we do not propagate that field from \$_GET[]. Finally, those parameters can also occur in \$path_info, so we cleanup that string too.

If \$propagate is FALSE we don't propagate after all. However, in this case we do combine \$path and \$add to a single usable href.

- **TODO** Maybe we could insert the parameters from \$add into \$path_info. OTOH: adding the existing way always works. If it aint broken... (2014-10-09/PF)

bool function login_send_bypass(\$user) [line 858]

Function Parameters:

- *array* **\$user** an associative array with the user record

send a new (temporary) password to the user via email

This generates a new temporary password for the user, stores it in the user record and sends an email message to the user with the temporary password (in plain text) and further instructions.

Note that the password is valid only for a limited time; sending a password in plain text appears to be an acceptable risk. Note that the limited time is increased with 10% in order to give the user a reasonable margin to enter the correct password.

Also note that the existing salt is used to salt the temporary password; this makes it easier to check for validity of both the regular password and the temporary password later on.

A log message recording the event is added via [logger\(\)](#).

- Uses [logger\(\)](#)
- Uses \$CFG

bool function login_send_confirmation(\$user) [line 913]

Function Parameters:

- *array* **\$user** an associative array with the user record

send email to user confirming password change

This sends an email to the user's email address confirming that the user's password was changed. Note that the new password is **_NOT_** sent to the user.

- Uses \$CFG

bool function login_send_laissez_passer(\$user) [line 791]

Function Parameters:

- *array* **\$user** the user record from database

send a special one-time login code to the user via email

This generates a temporary code with which the user can request a new temporary password. This code can be used only once. Note that this code is valid for only a limited time. This code simply overwrites the bypass password (the temporary password) in the user record. This means that if a phase 2 is pending, a new phase 1 will replace the old phase 2.

The temporary code consists of digits and uppercase characters. However, it is longer (20 characters) than the minimum password length of 6, so a brute force on such a code will likely not succeed (36^{20} is much more than the usual 62^6).

This routine also brings the user's record into 'bypass mode'. This mode is reset to 'normal' after the user has successfully changed her password.

A log message recording the event is added via [logger\(\)](#).

- Uses [logger\(\)](#)
- Uses \$CFG

bool function password_hash_check(\$salt, \$password, \$hash) [line 1179]

Function Parameters:

- *string* **\$salt** salt
- *string* **\$password** password to check
- *string* **\$hash** hash to check against

check equivalency of salt+password against hash

This verifies whether the hash of \$salt and \$password is the same as \$hash. Note that the two hashes are compared in a caseInsensitive way. Usually these hashes are using lowercase hexadecimal digits but a caseInsensitive compare makes A,...,F equivalent to a,...,f.

If the length of the presented \$hash is 40 characters, it is assumed that the hash algorithm to use is sha1, otherwise the default algorithm (md5) is used.

- **TODO** Now if the sha1-hash was compressed into 5 or 6 bits/char instead of hexadecimal digits this routine would fail miserably, so don't do that.
- Uses [was_password_hash\(\)](#)

string function password_salt([\$length = 12]) [*line 1193*]

Function Parameters:

- *int* **\$length** the number of characters in the generated string

generate a quasi random string to salt the password hash

this generates a quasi-random string of digits and letters to be used as a salt when calculating a password hash.

void function show_login(\$dialog, [\$message = "], [\$propagate = TRUE]) [*line 625*]

Function Parameters:

- *int* **\$dialog** is the screen variant to show, could be 1,...,5
- *string* **\$message** the message to show just above the first field, used for feedback to user
- *bool* **\$propagate** TRUE means propagate the existing parameters, FALSE means don't propagate

show complete login dialog and exit

There are different variations of this dialog.

1. LOGIN_PROCEDURE_NORMAL Plain login

(message)

Username: _____

Password: _____

[OK]

<home page> <forgotten password?>

This screen is used for plain user authentication. As a rule the user uses the correct primary password to authenticate. However, it is also possible to enter the 'bypass' password instead. If the authentication fails, that fact is recorded. If the number of failures exceeds threshold N, the user is shown screen #3

LOGIN_PROCEDURE_SEND_LAISSEZ_PASSER. If the number is still below N screen #1 is shown again.

The link <forgotten password?> takes the user directly to screen #3 LOGIN_PROCEDURE_SEND_LAISSEZ_PASSER.

2. LOGIN_PROCEDURE_CHANGE_PASSWORD - Login/change password (message)

Username: _____

Old password: _____

New password1: _____

New password2: _____

[OK] This screen is used to change the user's password. If both new passwords are different, the user is redirected to the same screen #3 until she gets it right. Otherwise, if the old password is either the valid original password OR the bypass password, the password is changed and the mode is reset to 'normal'. The one-time codes and the bypass password are reset. Also, as a result, the user is logged in. If the user failed to enter the proper old password more than N times, the mode is also reset to normal (invalidating the laissez-passer and the bypass password) and the user is dropped at a screen basically telling her to contact the webmaster. In this process the user is also logged out if necessary.

3. LOGIN_PROCEDURE_SEND_LAISSEZ_PASSER Request bypass (message)

Username: _____

Email: _____

[OK] This screen is used to help the user reset her password. It is displayed automatically after N failed login attempts. This screen can also be reached via the <forgot password?> link in screen #1.

If the user presents an invalid combination of username and email address, this failure is also recorded. If the number of failures has reached the threshold N, the user is taken to a screen that basically tells the user ask the webmaster for assistance and that's that.

If the user presents a valid combination of username and email address, an email with a message like 'click the link below for a new password' is sent to the email address. After that mail is sent a screen is displayed, basically telling the user to await further instructions that were sent via mail.

Note that resetting the password is a two-step process. First the user is sent a one-time code laissez-passer embedded in a link. Clicking the link before it expires (after T2 minutes) yields a second email message containing a bypass password that can be used to login and subsequently change the primary password. After that both the laissez-passer and the bypass password are invalidated.

4. LOGIN_PROCEDURE_SEND_BYPASS - Send a temporary password

(message)

Username: _____

One-time code: _____

[OK] Phase 2 of the forgot password procedure.

5. LOGIN_PROCEDURE_MESSAGE_BOX Alert (message)

[OK]

This screen is user to communicate various messages to the user, e.g. 'check your mail for instructions', 'contact webmaster', etc.

Note: the `$message` receives special treatment. The actual message is prefixed with the word 'Message: ' and the combination is wrapped within a B-tag and a SPAN. By carefully changing the style of those tags, we end up either with the bare `$message` with a yellow background if style is ON in the user's browser, OR 'Message: `$message` in bold face if the style is OFF. This gives good results in a text browser, eg. lynx.

void *int* function was_login([`$procedure` = LOGIN_PROCEDURE_SHOWLOGIN], [`$message` = ""]) [line 282]

Function Parameters:

- *int* **`$procedure`** the login procedure to execute
- *string* **`$message`** the message to display when showing the login dialog

execute the selected login procedure

The login process is controlled via the parameter 'login' provided by the user via 'index.php?login=N or via the 'action' property in a HTML-form. These numbers correspond to the LOGIN_PROCEDURE_* constants defined near the top of this file. Here's a reminder:

1. LOGIN_PROCEDURE_NORMAL this is the usual procedure for logging in
2. LOGIN_PROCEDURE_CHANGE_PASSWORD this is the procedure to change the user's password
3. LOGIN_PROCEDURE_SEND_LAISSEZ_PASSER this is phase 1 of the 'forgot password' procedure
4. LOGIN_PROCEDURE_SEND_BYPASS this is phase 2 of the 'forgot password' procedure

It is also possible that code in [main_index.php](#) or [main_admin.php](#) calls this routine without the login=N parameter, e.g. when a user attempts to access a page in an intranet without being logged in already.

Note that this routine only returns to the caller after either a succesful regular login (i.e. after completing LOGIN_PROCEDURE_NORMAL). All the other variants and error conditions yield another screen and an immediate exit and hence no return to caller. If this routine returns, it returns the user_id of the authenticated user (the primary key into the users table). It is up to the caller to retrieve additional information about this user; any information read from the database during login is discarded. This prevents password hashes still lying around.

Note that a successful login has the side effect of garbage collection: whenever we experience a successful login any obsolete sessions are removed. This makes sure that locked records eventually will be unlocked, once the corresponding session no longer exists.

The garbage collection routine is also called from the PHP session handler every once in a while, but here we make 100% sure that garbage is collected at least at every login. (Note: obsolete sessions should not be a problem for visitors that are not logged in, because you have to be logged in to be able to lock a record.)

Note that in `LOGIN_PROCEDURE_SEND_BYPASS` we refrain from propagating the parameters. because the user might arrive here with `GET['code']` and `GET['username']`. We don't want to propagate that. Side effect is that nothing is propagated, but what's the point anyway when you are busy resetting your password. Not having parameters propagate is not important at that time.

- Uses [dbsession_setup\(\)](#)
- Uses [dbsession_garbage_collection\(\)](#)
- Uses `$CFG`

void function was_logout([\$redirect_flag = TRUE], [\$procedure = LOGIN_PROCEDURE_NORMAL]) [line 189]

Function Parameters:

- *bool* **\$redirect_flag** TRUE redirects to `$_SESSION['redirect']`, FALSE yields login dialog
- *int* **\$procedure** determines which login dialog to show

end a session (logout the user) and maybe redirect

This routine ends the current session if it exists (as indicated by the cookie presented by the user's browser). An empty value is sent to the browser (effectively deleting the cookie) and also the session is ended. The routine ends either with showing a generic login dialog OR a redirection to a user-defined page.

Note that as a rule this routine does NOT return but instead calls `exit()`. However, there are cases where this routine DOES return, notably when no session appears to be established (no cookie submitted by the browser. If the routine does return, the status is equivalent to a logged out user; no session exists so the user simply should not be logged in.

On successful logout, the user usually is redirected to her own redirect-page. Thos depends on the `$redirect_flag`: if it is FALSE we always force a normal login dialog even if `$redirect` is specified. This is used from [was_login\(\)](#) to make sure that we end in a login dialog after (forcefully) logging out the user. This also gets rid of the 'logout=' parameter preventing a logout loop.

- Uses [dbsession_setup\(\)](#)
- Uses \$CFG

string function was_password_hash(\$salt, \$password, [\$algorithm = 0]) [*line 1149*]

Function Parameters:

- *string* **\$salt**
- *string* **\$password**
- *int* **\$algorithm** (optional) algorithm to use: 0=md5, 1=sha1

calculate a hash from a salt and a password

This routine constructs a hash of the combination of salt and password. By default the md5() function is used to calculate a 32-character long string of hexadecimal digits. If the parameter \$algorithm is 1 then the sha1() function is used and a 40-character long string of hexadecimal digits is returned.

Note that we do not use the crypt() function because that could introduce a portability issue. If a website is migrated to another machine, the used crypt algorithm might no longer be available, and that would effectively lock out all users. Both md5() and sha1() are standard PHP-functions (since 4.3.x) and should be portable, which makes any installed table of users portable too.

Note that this routine used to be called 'password_hash'. PHP 5.5.0 introduced a function called 'password_hash' which clashed with our own version, hence we renamed this function to was_password_hash().

- Usedby [password_hash_check\(\)](#)

modulemanagerlib.php

/program/lib/modulemanagerlib.php - modulemanager

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: modulemanagerlib.php,v 1.10 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

TASK_MODULEMANAGER_EDIT = edit [line 29]
TASK_MODULEMANAGER_INTRO = intro [line 27]
TASK_MODULEMANAGER_SAVE = save [line 28]
void function job_modulemanager(&\$output) [line 40]

Function Parameters:

- *object* **&\$output** collects the html output

main entry point for modulemanager (called from /program/main_admin.php)

this routine dispatches the tasks, If the specified task is not recognised, the default task TASK_MODULEMANAGER_INTRO is executed.

void function modulemanager_cmp(\$a, \$b) [line 153]

Function Parameters:

- *array* **\$a**
- *array* **\$b** \$return int indicating the ordering of \$a and \$b like strcmp()

compare two arrays by the title member (for sorting modules)

array function modulemanager_get_modules([\$forced = FALSE]) [line 120]

Function Parameters:

- **bool \$forced** if TRUE a fresh trip to the database is forced

retrieve a list of modules that should appear in the module manager

this routine returns an array with id, name, title and description of all active modules that have at least one parameter in the modules_properties table. If there are no modules available (or an error occurs) an empty array is returned. The modules in the list is ordered by the translated name (title) of the module, i.e. the order depends on the current translation language.

void function modulemanager_process(&\$output, \$task) [line 178]

Function Parameters:

- **object &\$output** collects the html output
- **\$task**

handle the editing/saving of the main configuration information

this routine handles editing of the main configuration parameters. It either displays the edit dialog or saves the modified data and shows the configuration manager introduction screen.

Note that we do NOT try to redirect the user via a header() after a succesful save. It would be handy because this particular save action may have had impact on the global configuration, which is already read at this point. By redirecting we would make a fresh start, with the new parameters. However, we lose the easy ability to tell the user that the data was saved (via \$output->add_message()). So, either no feedback or obsolete global config in core. Hmmmm. I settle for the feedback and the 'wrong' settings.

- **Uses** ConfigAssistant()

void function modulemanager_show_intro(&\$output) [line 71]

Function Parameters:

- **object &\$output** collects the html output

display an introductory text for the module manager

void function modulemanager_show_menu(&\$output, [\$current_module = NULL]) [*line 83*]

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **\$current_module** indicates the current menu selection (if any)

display the module manager menu

pagemanager.class.php

/program/lib/pagemanager.class.php - pagemanager

This file contains the Page Manager class, the core functionality of Website@School.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: pagemanager.class.php,v 1.41 2016/05/26 13:22:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

DIALOG_NODE_ADD = 1 *[line 58]*
DIALOG_NODE_DELETE_CONFIRM = 5 *[line 62]*
DIALOG_NODE_EDIT = 2 *[line 59]*
DIALOG_NODE_EDIT_ADVANCED = 3 *[line 60]*
DIALOG_NODE_EDIT_CONTENT = 4 *[line 61]*
MODULE_NAME_DEFAULT = htmlpage *[line 73]*

Default initial module of a new page (see get_dialogdef_add_node())

NODE_VISIBILIY_DEFAULT = NODE_VISIBILIY_HIDDEN *[line 71]*

Default initial visibility of a new node (see get_dialogdef_add_node())

NODE_VISIBILIY_EMBARGO = 3 *[line 69]*

Initial visibility of a new node: under embargo

NODE_VISIBILIY_HIDDEN = 2 *[line 67]*

Initial visibility of a new node: hidden

NODE_VISIBILIY_VISIBLE = 1 *[line 65]*

Initial visibility of a new node: visible

PARAM_SUBMENU_OPTION = option *[line 55]*
PARAM_TREEVIEW = treeview *[line 49]*
TASK_ADD_PAGE = addpage *[line 35]*
TASK_ADD_SECTION = addsection *[line 36]*
TASK_NODE_DELETE = delete *[line 37]*
TASK_NODE_EDIT = edit *[line 38]*

TASK_NODE_EDIT_ADVANCED = editadvanced [line 40]
TASK_NODE_EDIT_CONTENT = editcontent [line 39]
TASK_NODE_FORCE_UNLOCK = unlock [line 46]
TASK_PAGE_PREVIEW = preview [line 41]
TASK_SAVE_CONTENT = savecontent [line 45]
TASK_SAVE_NEWPAGE = savenewpage [line 43]
TASK_SAVE_NEWSECTION = savenewsection [line 44]
TASK_SAVE_NODE = savenode [line 42]
TASK_SET_DEFAULT = setdefault [line 34]
TASK_SUBTREE_COLLAPSE = collapse [line 33]
TASK_SUBTREE_EXPAND = expand [line 32]
TASK_TREEVIEW = treeview [line 30]
TASK_TREEVIEW_SET = settreeview [line 31]
TREE_VIEW_CUSTOM = 2 [line 51]
TREE_VIEW_MAXIMAL = 3 [line 52]
TREE_VIEW_MINIMAL = 1 [line 50]

statisticslib.php

/program/lib/statisticslib.php - statistics

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: statisticslib.php,v 1.9 2016/06/15 12:11:41 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

void function job_statistics(&\$output) [*line 41*]

Function Parameters:

- *object* **&\$output** collects the html output

main entry point for statistics (called from admin.php)

Produce a simple ordered table of pages with # of page views for one area or for all areas the user is allowed to access.

Note: Even though expired pages are no longer visible for visitors, the # of views is still displayed. Pages under embargo are completely suppressed.

- **TODO** this routine is a little bit too long, it should be split into smaller chunks

bool function statistics_embargo(&\$sections, &\$now, \$id) [*line 226*]

Function Parameters:

- *array* **&\$sections** holds pertinent information about sections in selected areas
- *string* **&\$now** current time to check against the embargo time

- *int* **\$id** section id

check ancestors for embargo

ascend the tree via \$parent_id to see if there is an embargo. the quest ends when we reach a top level section (indicated by \$id == \$parent_id) or a section that was examined before (indicated through the existence of the flag element in the \$section). before we start recursing we set that flag just to be sure we don't end up in an endless loop if somehow the tree structure is not sane (contains circular references).

bool|array function statistics_get_pages(&\$areas, \$area_id) [line 261]

Function Parameters:

- *array* **&\$areas** list of areas available to the current user
- *int|null* **\$area_id** area of interest or NULL indicating all areas in \$areas[]

construct an ordered list of pages to show in a statistics report

strategy:

- start with all pages in selected area(s) NOT under embargo
 - retrieve all sections in selected areas(s)
 - for all pages: if none of the ancestors is under embargo then keep page in \$pages[]
- If anything goes wrong, return FALSE, otherwise an array with the selected pages

It would be nice if we could simply select \$limit records from the database at \$offset. Unfortunately we cannot be sure if a page is eligible for display until we have examined the ancestors so we have to retrieve them and perhaps later discard them. oh well, there's room for improvement.

void function statistics_show_area_menu(&\$output, &\$areas, [\$current_area_id = NULL]) [line 176]

Function Parameters:

- *object* **&\$output** collects the html output
- *array* **&\$areas** list of all areas available for this user
- *int|null* **\$current_area_id** the current area

show a menu of available areas

output a menu, starting with an entry to show _all_ available areas, followed by links to individual areas. If there is only a single area, the 'all areas' link is suppressed

theme.class.php

/program/lib/theme.class.php - taking care of themes

This file defines a base class for dealing with themes. It is always included and it can be used as a starting point for other themes by inheriting from this class.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: theme.class.php,v 1.35 2016/05/26 13:22:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

themelib.php

/program/lib/themelib.php - theme factory

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: themelib.php,v 1.6 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool/object function theme_factory(\$theme_id, \$area_id, \$node_id) [*line 45*]

Function Parameters:

- *int* **\$theme_id** denotes which theme to retrieve from database via primary key
- *int* **\$area_id** the area we're working in
- *int* **\$node_id** the node that is to be displayed

manufacture a theme object

This loads (includes) a specific theme based on the parameter \$theme_id. Relevant data is read from the database.

- **TODO** what if the theme is not found? Currently no alternative is loaded but FALSE is returned.
- **TODO** should we massage the directory and file names of the included theme?
- **Uses** \$CFG

require_once **\$CFG->progrdir."/lib/theme.class.php"** [*line 29*]

theme class is used as a base class from which others can be derived

tokenlib.php

/program/lib/tokenlib.php - functions to manipulate unique tokens via the database

This file provides the functions to manipulate tokens stored in the database. These tokens are used to create unique instances of dialogs (forms) making it impossible to POST data to a form without first retrieving the individual (blank) form to start with.

Tokens are stored in a table called 'tokens' which is defined as follows: +-----+-----

Field	Type	Null	Key	Default	Extra
token_id	int(11)		PRI	NULL	auto_increment
token_key	varchar(60)		MUL		
token_ref	varchar(60)				
token_start	int(11)			0	
token_end	int(11)			0	
token_expire	int(11)			0	
remote_addr	varchar(150)				
data	longtext	YES		NULL	

Note: this table was added in v0.90.5 (June 2013), initially for the mailpage module.

The following functions are defined. \$token_id = token_create(\$ref, &\$token_key, \$delay_start, \$delay_end, \$ip_addr);
\$token_id = token_lookup(\$ref, \$token_key, &\$timer_start, &\$timer_end, &\$ip_addr, &\$data);
\$retval = token_store(\$token_id, \$data);
\$retval = token_fetch(\$token_id, &\$data);
\$retval = token_destroy(\$token_id);
\$retval = token_garbage_collect();

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: tokenlib.php,v 1.7 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool/int function token_create(\$reference, &\$token_key, [\$delay_start = 10], [\$delay_end = 14400], [\$ip_addr = NULL]) [*line 102*]

Function Parameters:

- *string* **\$reference** is an identifier of the dialog requesting a token
- *string* **&\$token_key** a generated unique identifier based on the pkey of the token
- *int* **\$delay_start** seconds to wait before POST'ed data will be considered valid
- *int* **\$delay_end** seconds after which the dialog no longer accepts POST'ed data
- *string* **\$ip_addr** the IP-address this visitor is calling from

create a new record in the tokens table, return the unique token_id

this creates generates a new unique token_key and stores it in a new record in the tokens table. Additional information is recorded in the new record too:

- the dialog/form reference
- the earliest unix timestamp POST'ed data will be accepted (based on \$delay_start)
- the latest unix timestamp POST'ed data will be accepted (based on \$delay_end)
- the unix timestamp after which this record can be deleted (via garbage collection)
- the IP-address of the current caller (from \$_SERVER['REMOTE_ADDR'])

This routine creates a unique token_key. This key consists of the hexadecimal representation of the token_id (which is the unique serial of the record) followed by a quasi-random string of hexadecimal characters. There is no way there will be a repeat ever, unless the serial wraps round. I assume (dangerous, I know) that this is very unlikely to happen any time soon. In order to further obfuscate the generated key the (sequential) serial number is xor'ed with 878133331 before the random string is appended.

The parameter \$reference is used to further limit the chance that POST'ing a rogue token_key could influence another, valid \$token_key. This \$reference could be a number identifying a particular dialog definition, e.g. by using the filename and a line number in a define().

Finally, the IP-address of the visitor is recorded. This can be used in the future to limit brute force attacks on a form. Currently it is not used other than to add to the mail that is sent via the mailpage module. Note that by default we use the IP-address from \$_SERVER but the caller is free to substitute something else, perhaps a canonical IPv6-address.

We call the garbage collector from here. This keeps the table clean. Furthermore, we have got the time: the user has to wait at least \$delay_start seconds before she can submit anything, so I think there is no rush. YMMV.

- **TODO** Should we enforce valid UTF8 in \$reference and \$ip_addr? We might have substr() trouble...

bool function token_destroy(\$token_id) [line 231]

Function Parameters:

- *string* **\$token_id** the unique token_id (pkey) that identifies the record

remove a token record from the tokens table (it should still exist)

remove the specified record from the table. it is an error if the record does not exist.

bool function token_fetch(\$token_id, &\$data) [line 195]

Function Parameters:

- *int* **\$token_id** the unique token_id (pkey) that identifies the token record
- *mixed* **&\$data** receives the unserialised data from the database

retrieve the (unserialised) data from the database

bool function token_garbage_collect() [line 260]

remove all expired tokens

this removes all records that are expired. Since this routine is also called from token_create() we have a big chance to keep the tokens table clean. However: logging every delete() may be a little too much so it is commented out.

This way, the worst that can happen is that someone keeps GET'ting a form that uses tokens every second for 8 hours in a row and subsequently nobody ever visits that form again. In that case we eventually have $8 \times 3600 \times 1 = 28800$ garbage records. Oh well.

OTOH: with all those crawlers and spiders on the WWW there is bound to be a 'bot' visiting that form the next day, effectively cleaning up for us. (Robots are a feature, not a bug...)

- **TODO** This routine should be called from cron.php every once in a while

bool function token_lookup(\$reference, \$token_key, &\$token_start, &\$token_end, &\$remote_addr, &\$data) [line

Function Parameters:

- *string* **\$reference** is an identifier of the dialog requesting a token
- *string* **\$token_key** a unique identifier based on the pkey of the token
- *int* **&\$token_start** unix timestamp indicating start of the valid interval
- *int* **&\$token_end** unix timestamp indicating end of the valid interval
- *string* **&\$remote_addr** the IP-address this visitor that created this token
- *mixed* **&\$data** receives the unserialised data from the database

lookup \$reference + \$token_key in the table and retrieve token information

This checks the existence of a token record in the tokens table. This \$token_key can only generated via [token_create\(\)](#). Furthermore, we require that the \$reference matches the one in the record. This prevents us accepting spurious token keys via manipulated requests. If the key does not exist, the call fails and FALSE is returned, otherwise the token_id is returned + data in the other parameters (timers, remote_addr, etc).

Typical use:

```
if (($token_id = token_lookup($ref, $key, $t0, $t1, $ip_addr, $data)) === FALSE) {
    logger('error: no such token');
} elseif (time() < $t0) {
    logger('error: whoa, not so fast there!');
} elseif ($t1 < time()) {
    logger('error: too late');
} else {
    logger('welcome visitor from ' . $ip_addr);
}
```

This allows for accepting POST'ed information only within the time window defined by \$t0 and \$t1 and denying access otherwise with a precise cause (no token, too early, too late).

bool function token_store(\$token_id, \$data) [line 215]

Function Parameters:

- *int* **\$token_id** the unique token_id (pkey) that identifies the token record
- *mixed* **\$data** holds the unserialised data to store

write the (serialised) data to the database
serialise and store \$data in the database

toolslib.php

/program/lib/toolslib.php - tools

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: toolslib.php,v 1.14 2016/06/28 14:08:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

CHORE_SESSION_DELETE = delete [line 34]
CHORE_SESSION_VIEW = view [line 33]
TASK_BACKUPTOOL = backuptool [line 31]
TASK_LOGVIEW = logview [line 35]
TASK_SESSIONTOOL = sessiontool [line 32]
TASK_TOOLS_INTRO = intro [line 29]
TASK_TRANSLATETOOL = translatetool [line 30]
void function job_tools(&\$output) [line 47]

Function Parameters:

- *object* **&\$output** collects the html output

main entry point for tools (called from /program/main_admin.php)

this routine dispatches the tasks, If the specified task is not recognised, the default task TASK_TOOLS_INTRO is executed.

- **TODO** fix permissions for backup tool! perhaps another bit?

void function logview_download(&\$output, \$pivot) [line 808]

Function Parameters:

- *object* **&\$output** collects output to show to user

- *int* **\$pivot** download limit: pkey must be < \$pivot

download records from the log table in CSV format

this sends the contents of the logtable upto (but excluding) the record with pkey = \$pivot to the user in CSV format

the suggested filename contains the name of the host and the first and the last pkey of the records exported. This allows for downloading logs from different hosts and multiple times from the same host without name clashes.

If there are errors or no records nothing is sent to the user (but we do try to log the event).

Note The number Nr displayed in the overview (see [logview_show\(\)](#)) is just a number starting at 1 every time. The download file works with the real record pkey.

string function logview_priority(\$p) [*line 982*]

Function Parameters:

- *int* **\$p** numeric value priority

helper routine to translate a numeric priority to human readable text

look up the symbolic name of priority \$p

[Rant]

I used to use the built-in constants like LOG_INFO and LOG_DEBUG to allow for different levels of logging (see [logger\(\)](#)). To my complete surprise logging didn't work at all on Windows (it did on Linux). The reason was that LOG_DEBUG and LOG_INFO and LOG_NOTICE are all defined to be the same value. WTF? Any test based on LOG_DEBUG and LOG_INFO being different would fail, hence no logging at all. The mind boggles! So, instead of using built-in constants I had to define my own and do a global search&replace. Aaarghhhhh!!!!

[/Rant]

void function logview_prune(&\$output, \$pivot) [*line 866*]

Function Parameters:

- *object* **&\$output** collects output to show to user
- *int* **\$pivot** download limit: pkey must be < \$pivot

ask for confirmation and/or maybe execute pruning of log messages table

shows confirmation screen including download link and also actually process the deletions if the user confirmed. We give the user another opportunity to download the selected records before pruning.

void function logview_show(&\$output, [\$offset = NULL]) [line 671]

Function Parameters:

- *object* **&\$output** collects output to show to user
- *null|int* **\$offset** indicates where to start the listing, NULL=consult GET parameters

show entries from log table in a neat HTML-table (possibly paginated)

this constructs a table of the HTML-variety with the contents of the logtable. fields displayed are: datim, IP-address, username, logpriority and message we use a LEFT JOIN in order to get to a meaningful username rather than a numeric user_id an attempt is made to start with the last page of the logs because that would probably be the most interesting part. We paginate the log in order to keep it manageable.

void function sessions_delete(&\$output, \$session_id) [line 531]

Function Parameters:

- *object* **&\$output** collects the html output
- *int* **\$session_id**

handle confirmation and actual delete of a session

this single routine handles

- the display of a confirmation message,
- handling of the actual delete, and
- handling of the bail out/cancelbutton

mixed function sessions_show(&\$output, [\$id = NULL], [\$show_delete = TRUE]) [line 343]

Function Parameters:

- *object* **&\$output** collects the html output
- *mixed* **\$id** select which session(s) to display (Null means all)
- *bool* **\$show_delete** if TRUE show clickable links to delete option

show a table with sessions

display a table with sessions and locked nodes per session

If \$id is NULL, all current sessions+locks are displayed, if \$id is a specific session_id only that session is displayed. (used in the session delete chore, see [sessions_delete\(\)](#)).

If \$show_delete is TRUE a link is displayed that allows the user to delete a session, if FALSE no link is displayed.

void function sessions_show_footer(&\$output) [*line 514*]

Function Parameters:

- *object* **&\$output** collects the html output

close the HTML-table

void function sessions_show_header(&\$output, &\$class, [\$show_delete = TRUE]) [*line 397*]

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **&\$class** attribute of table rows and cells
- *bool* **\$show_delete** if TRUE show clickable links to delete option

show the header of the HTML-table

D	sss	ctime	atime	username (full_name)	ip_addr
	nnn	ltime	link_text (title)		

the HTML-table has 5 or 6 columns, depending on \$show_delete. the header takes two rows: one for the session information, and another for the node lock information.

Parameter \$class is used for styling of the table header.

void function sessions_show_node_locked(&\$output, &\$record, &\$class, [\$show_delete = TRUE]) [line 489]

Function Parameters:

- *object* **&\$output** collects the html output
- *array* **&\$record** contains session and lock information
- *string* **&\$class** attribute of table rows and cells
- *bool* **\$show_delete** if TRUE show clickable links to delete option

show locked node information from a single (combined) record

the HTML-table has 5 or 6 columns, depending on \$show_delete. Parameter \$class is used for styling of the table row. The value is toggled between 'odd' and 'even'.

void function sessions_show_overview(&\$output) [line 320]

Function Parameters:

- *object* **&\$output** collects the html output

show an overview of currently existing sessions

void function sessions_show_session(&\$output, &\$record, &\$class, [\$show_delete = TRUE]) [line 436]

Function Parameters:

- *object* **&\$output** collects the html output
- *array* **&\$record** contains session and lock information
- *string* **&\$class** attribute of table rows and cells
- *bool* **\$show_delete** if TRUE show clickable links to delete option

show session information from a single (combined) record

the HTML-table has 5 or 6 columns, depending on \$show_delete. Parameter \$class is used for styling of the table row. The value is toggled between 'odd' and 'even'.

void function show_tools_intro(&\$output) [line 123]

Function Parameters:

- *object* **&\$output** collects the html output

display an introductory text for tools + menu

void function show_tools_menu(&\$output, [\$current_task = NULL]) [*line 135*]

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **\$current_task** indicate the current menu selection (if any)

display the tools menu

output function task_backuptool(&\$output) [*line 218*]

Function Parameters:

- *object* **&\$output** collects output to show to user

show an introductory text for backup tool OR stream a ZIP-file to the browser

If we arrive here via the tools menu, the parameter download is not set. In that case we show an introductory text with a link that yields a ZIP-file with the backup.

If the user follows the download link, we arrive here too but with the download parameter set. We then dump the database in a variable and subsequently compress it in a ZIP-file which we stream to the browser. We do code some things in the basename of the backup:

- the hostname
- the database name
- the database prefix
- the date and the time

which should be enough to distinguish nearly all backups if you happen to have a lot of different ones. Note that the URL is also encoded as a comment in the .ZIP.

The parameter download is currently set to 'zip'. However, we do attempt to send the plain uncompressed data if that parameter is set to 'sql' (quick and dirty). Oh well. hopefully there is enough memory to accomodate backups of moderate sized sites.

Note that we need space to compress the data; a informal test yielded that we need about 160% of the uncompressed size of the backup (tested with a small testset). Rule of the thumb for memory: the more the merrier but at least twice the size of the uncompressed backup.

void function task_logview(&\$output) [line 646]

Function Parameters:

- *object* **&\$output** collects output to show to user

quick and dirty logfile view / download / prune tool

this is the dispatcher that decides whether to show entries, download entries or prune the logs

- **TODO** should we allow for fancy selection mechanisms on the logfile or is that over the top?

void function task_sessiontool(&\$output) [line 290]

Function Parameters:

- *object* **&\$output** collects the html output

main entry point for session management

string function tools_get_hostname() [line 949]

helper routine to construct something that looks like this hosts name

translatetool.class.php

/program/lib/translatetool.class.php - taking care of language translations

This file defines a class for managing translations.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: translatetool.class.php,v 1.17 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
TRANSLATETOOL_CHORE_EDIT = edit [line 34]
TRANSLATETOOL_CHORE_LANGUAGE_ADD = language_add [line 30]
TRANSLATETOOL_CHORE_LANGUAGE_EDIT = language_edit [line 32]
TRANSLATETOOL_CHORE_LANGUAGE_SAVE = language_save [line 33]
TRANSLATETOOL_CHORE_LANGUAGE_SAVE_NEW = language_savenew [line 31]
TRANSLATETOOL_CHORE_OVERVIEW = overview [line 29]
TRANSLATETOOL_CHORE_SAVE = save [line 35]
TRANSLATETOOL_PARAM_DOMAIN = domain [line 39]
TRANSLATETOOL_PARAM_LANGUAGE_KEY = language_key [line 38]
```

This parameter identifies the language.

Note: it should not be confused with the global 'language' parameter

updatelib.php

/program/lib/updatelib.php - update wizard

This file handles all system updates. The basic idea is as follows.

We assume that a previous version of Website@School is/was already correctly installed using the code in /program/[install.php](#). Using this version the school has added many, many hours of work in entering data.

Now a new version is installed, i.e. the new files (or just the updated files) are copied/uploaded to the webserver, including the file [version.php](#) and perhaps also updated versions of modules, themes, etc. This yields an error message for visitors (the infamous 'error 050') because the database version and the file version no longer match. The user logging in into admin.php is forced to attend to the job 'update', arriving here in [job_update\(\)](#).

There an overview is presented of the core version and versions of all subsystems, with the option to upgrade those that qualify. The actual work for the core version is done in [update_core\(\)](#) in this file. The actual work for modules and themes are done via the code in those subsystems. However, these are called from here.

The user is more or less _forced_ to perform the upgrade: all ways lead to job_upgrade() while the internal (database) version does not match the file version.

The upgrade should perform all necessary steps to upgrade, ending with updating the internal version number to match the file version. After that the error message '050' for visitors is gone, and admin.php no longer forces the user to come here. It is possible, however, to manually arrive here to check the updates of modules, themes, etc. but I consider that less important. That is: an upgrade of a module or theme will NOT be forced upon the user. It is wise, though to upgrade, but that is up to the user.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: updatelib.php,v 1.43 2016/06/28 14:36:55 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
TASK_INSTALL_LANGUAGE = install_language [line 66]
TASK_INSTALL_MODULE = install_module [line 69]
TASK_INSTALL_THEME = install_theme [line 72]
TASK_UPDATE_CORE = core [line 64]
```

```
TASK_UPDATE_LANGUAGE = update_language [line 67]
TASK_UPDATE_MODULE = update_module [line 70]
TASK_UPDATE_OVERVIEW = overview [line 63]
TASK_UPDATE_THEME = update_theme [line 73]
array function get_manifests($path) [line 1048]
```

Function Parameters:

- *string* **\$path** top directory for the search for manifest files

retrieve an array of manifests for modules, themes or languages

this examines the file system starting in the directory \$path, looking for manifest files. These manifest files are named after the subdirectory they are in as follows. Example: If \$path is /program/modules, this routine steps through that directory and may find subdirectories 'htmlpage', 'guestbook' and 'forum'. Eventually these manifest files are include()'d:

```
                                /program/modules/htmlpage/htmlpage_manifest.php,
/program/modules/guestbook/guestbook_manifest.php                        and
/program/modules/forum/forum_manifest.php.
```

Every manifest file must describe the module (or language or theme) via the following construct:

```
$manifests['htmlpage'] = array('name' => 'htmlpage', ..., 'cron_interval' => 0);
```

After processing all the subdirectories of \$path, the resulting array \$manifests is returned. Note that pseudo-directories like '.' and '..' are not considered. Also, subdirectories 'foo' without the file 'foo_manifest.php' are also ignored.

Note that the name of the manifest file itself is also stored in the array, but excluding the subdirectory name.

Note: a similar routine is used in the installation script [install.php](#).

```
void function install_language(&$output, $language_key, $language_key) [line 343]
```

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **\$language_key** primary key for language record in database AND name of the /program/languages subdirectory
- **\$language_key**

install an additional language pack

this routine attempts to insert the information from the manifest of language \$language_key into the database. The routine displays the result (error or success) in a message in \$output. Details can be found in the logs.

The `language_key` is validated by reading all existing manifests. This is quite expensive, but that is not important because we do not use this routine very often anyway.

Note that we assume that the actual translations of the language pack are already unpacked into the correct directories. The corresponding manifest should exist in the directory `/program/languages/$language_key`.

`void function install_module(&$output, $module_key) [line 475]`

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **\$module_key** primary key for module record in database AND name of the `/program/modules` subdirectory

install an additional module

this routine attempts to insert the information from the manifest of module `$module_key` into the database. The routine displays the result (error or success) in a message in `$output`. Details can be found in the logs.

The `module_key` is validated by reading all existing module manifests. This is quite expensive, but that is not important because we do not use this routine very often anyway.

Note that we assume that the actual modules are already unpacked into the correct directories. The corresponding manifest should exist in the directory `/program/modules/$module_key`.

- **TODO** we should refactor and combine `install_theme()` and `install_module()`
- **Uses** `$CFG`

`void function install_theme(&$output, $theme_key) [line 696]`

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **\$theme_key** primary key for theme record in database AND name of the `/program/themes` subdirectory

install an additional theme

this routine attempts to insert the information from the manifest of theme \$theme_key into the database. The routine displays the result (error or success) in a message in \$output. Details can be found in the logs.

The theme_key is validated by reading all existing module manifests. This is quite expensive, but that is not important because we do not use this routine very often anyway.

Note that we assume that the actual theses are already unpacked into the correct directories. The corresponding manifest should exist in the directory /program/themes/\$theme_key.

- **TODO** we should refactor and combine install_theme() and install_module()
- **Uses** \$CFG

void function job_update(&\$output) [line 103]

Function Parameters:

- *object* **&\$output** collects the html output

main entry point for update wizard (called from /program/main_admin.php)

This routine takes care of executing update routines for both the core program and modules, themes, etc. It is called automatically whenever the core program version in the database is different from the version in the file {@Ink version.php} (see also [main_admin\(\)](#)).

It can also be called manually via 'job=update'. When no specific task is specified, this routine shows the overview of versions for core, modules, themes, etc. Whenever a component is NOT up to date, an [Update] button is displayed. If a component IS up to date, we simply display the word 'OK'. This implies that when everything is up to date, the overview simply displays a list of OK's and the user is 'free to go'.

The actual updates for modules, themes, etc. is done via the various subsystems themselves, e.g. by calling `htmlpage_upgrade()` in the file `/program/modules/htmlpage/htmlpage_install.php`. The updates for the core program are actually performed from this file right here, see below for an example.

Note that we give a core update high priority: if the core is not up to date, nothing will work, except updating the core.

void function update_core(&\$output) [*line 891*]

Function Parameters:

- *object* **&\$output** collects the html output

update the core version in the database to the version in the version.php file (the 'manifest' version)

bool function update_core_2010120800(&\$output) [*line 1119*]

Function Parameters:

- *object* **&\$output** collects the html output

perform actual update to version 2010120800

bool function update_core_2010122100(&\$output) [*line 1137*]

Function Parameters:

- *object* **&\$output** collects the html output

perform actual update to version 2010122100

bool function update_core_2011020100(&\$output) [*line 1155*]

Function Parameters:

- *object* **&\$output** collects the html output

perform actual update to version 2011020100

bool function update_core_2011051100(&\$output) [*line 1218*]

Function Parameters:

- *object* **&\$output** collects the html output

perform actual update to version 2011051100

this is a substantial change in the database: we (finally) standardise on UTF-8 including the database. Up until now we still only have a choice of exactly one database driver: MySQL. Therefore the upgrade we do here can be more or less MySQL-specific. (So much for database-independency).

What needs to be done here?

The most important task (in fact: the only task) is to change the collation (and implicitly the default charset) to utf8_unicode_ci (4.1.x <= MySQL < 5.5.2) or utf8mb4_unicode_ci (MySQL 5.5.3+). See [mysql.class.php](#) for more information on these UTF-8 & MySQL issues.

Strategy here is as follows.

```
for all 'our' tables (ie. "LIKE '{$prefix}%" do
  if table default charset is already utf8 (or utf8mb4)
    continue;
  for appropriate columns in this table
    change column type to binary
    change column type back to non-binary with correct charset and collation
  if no trouble sofar
    change default charset/collation of the table too
  else
    return failure
return success
```

This way we might be able to work our way through huge tables: if the PHP max processing time kicks in, we can rerun the upgrade and start (again) with the table we had in our hands the previous time. I don't expect this to happen, but it still the way to do it IMHO.

Note that I assume that I cannot change the default charset of the DATABASE for the same reason the Installation Wizard expects the database to be ready before installation commences. (I cannot be sure that I have the privilege to execute 'ALTER DATABASE \$db_name DEFAULT CHARSET utf8 COLLATE utf8_unicode_ci').

A useful reference for solving this problem of converting to utf8 can be found here:
http://codex.wordpress.org/Converting_Database_Character_Sets.

In the case of W@S we do not have to deal with enum-fields because those are not used at this time. In fact it boils down to changing char, varchar, text and longtext.

Let goforit...

bool function update_core_2011093000(&\$output) [line 1547]

Function Parameters:

- *object &\$output* collects the html output

perform actual update to version 2011093000

this is yet another substantial change in the database: after we (finally) standardised on UTF-8 the last time (see [update_core_2011051100\(\)](#)) a number of problems occurred with

new installations.

This specifically occurs with MySQL (currently the only supported database). In all their wisdom Oracle decided to change the default database engine from MyISAM to InnoDB in MySQL version 5.5.5. Bad move to do that somewhere in a sub-sub-release. Anyway. New installations with the default InnoDB engine AND with the 4-byte utf8mb4 character set (available since sub-sub-release 5.5.3) now generate serious trouble, because

- there is a hard-coded limit of 767 bytes for a key (index) in InnoDB, and
- every utf8mb4 character counts as four bytes never mind the actual content.

Note: the limit of 767 bytes stems from a utf8 (or utf8mb3 as it is now called) string of max. 255 characters and 1 16-bit string length. $255 * 3 + 2 = 767$ bytes. I wonder why UTF-8 wasn't implemented correctly (ie. with 1 to 4 bytes) to begin with and the key limit increased to $4 * 255 + 2 = 1022$ bytes. The limited UTF-8 support (only the BMP) now poses substantial problems. Yet another reason to start looking for an alternative database solution. BTW: the key limit in MyISAM is 1000 bytes.

These two conditions (InnoDB and utf8mb4) limit the length of a key (index) to 767 bytes / 4 bytes-per-char = 191 utf8mb4 characters. As it happens, some tables in WebsiteAtSchool used keyfields like varchar(240) and even varchar(255). These key sizes fail in InnoDB/utf8mb4 and the latter even fails with MyISAM/utf8mb4 because $255 * 4 + 2 = 1022$ bytes > 1000 bytes. What a mess...

So there you have it: all keys MUST be shortened to 191 characters max. in order to prevent stupid error messages about key too long. The alternative (forcing another character set such as 'ascii' or 'latin1' for some fields) doesn't cut it IMHO.

sigh

We still have a choice of exactly one database driver: MySQL. Therefore the upgrade we do here can be more or less MySQL-specific (so much for database-independency), as it has to be, because the syntax of ALTER TABLE is -- unsurprisingly -- MySQL-specific.

The good news is that we are still in beta, so a major change in the data definition is less painful than with hundreds of production servers...

Another issue is the use of foreign keys. We used to have a FK in the nodes tabledef along the lines of this construct: FOREIGN KEY parentnode (parent_id) REFERENCES nodes (node_id); Upto now this could not possibly have worked with InnoDB because adding a node would at the top level of an area would not satisfy this constraint. Since MyISAM silently ignores any foreign key definition it 'simply works' in that case. So, because this FK must be removed from earlier installations we need to DROP the FOREIGN KEY. However, since the whole program never installed using InnoDB, there is no need to drop this foreign key that wasn't even recorded (in a MyISAM database) in the first place. The same applies to a number of other FK's too: these are now removed from the various tabledefs but do no need to be DROPPed in this update routine.

What needs to be done here?

For existing tables some fields must be shortened from varchar(255) or varchar(240) to something like varchar(191) or even less. This **MUST** be done for key (index) fields. However, while we are at it some more fields **SHOULD** (or **COULD**) be shortened too. Here is what we do.

```
for all affected table.fields do
  if a record exists with current data length > proposed new length then
    tell the user about it
  endif
next
if there were data length errors then
  tell the user about manually fixing it
  bail out with result FALSE (= not upgraded)
endif
for all affected table.fields do
  change field definition to new length
  if errors
    tell the user about it (but carry on)
  endif
next
return results (TRUE on success, FALSE on 1 or more errors)
```

Below is a discussion of all affected fields and the rationale for picking the new lengths less than 191 characters.

```
config.name: varchar(240) => varchar(80)
modules_properties.name: varchar(240) => varchar(80)
themes_properties.name: varchar(240) => varchar(80)
themes_areas_properties.name: varchar(240) => varchar(80)
users_properties.name: varchar(240) => varchar(80)
users_properties.section: varchar(240) => varchar(80)
```

Currently the longest parameter name in use is 27 characters, so I have to admit that the arbitrary size of 240 is a little bit too much. I'll reduce these fields to a size of 80, which seems a little more realistic. As an additional bonus, this allows for a compound key using 'section' and 'name' in users_properties while staying within the limit of 767 bytes or 191 characters.

```
areas.path: varchar(240) => varchar(60)
groups.path: varchar(240) => varchar(60)
users.path: varchar(240) => varchar(60)
```

and

```
groups.groupname: varchar(255) => varchar(60)
users.username: varchar(255) => varchar(60)
```

The length of username or groupname was arbitrary set to 255. Different systems have different limits, e.g. 8, 14, 15, 16, 20, 32, 64 or 128. Since W@S is a stand-alone system we are more or less free to choose whatever we want (as long as it is less than 191 of course).

Since a username or groupname is only used to distinguish one user from another but at the same time giving at least some readability, a length of 255 is way too long. An arbitrary but hopefully more realistic choice is 60 characters.

The path for a user or group is derived from the corresponding name so it makes sense to make both fields the same length.

```
log_messages.remote_addr: varchar(255) => varchar(150)
login_failures.remote_addr: varchar(255) => varchar(150)
```

A remote address of type IPv4 generally looks like this: 'ddd.ddd.ddd.ddd' => length 15 It is not so easy to determine the length of an IPv6 address, because many valid variants exist.

'xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx' => length 39
'0000:0000:0000:0000:0000:0000:ddd.ddd.ddd.ddd' => length 45
'[0000:0000:0000:0000:0000:0000:ddd.ddd.ddd.ddd]' => length 47 (RFC3989)

[illegible]

Of course there several 'simplifications' such as omitting leading zeros in the hexquads and replacing the longest sequece of 0-hexquads with '::' that add to the confusion. RFC5952 adds the definition of a 'canonical representation' of IPv6 addresses to the party. Mmmm, see <http://xkcd.com/927>

My conclusion is: this whole IPv6-idea suffers from the Second System Syndrome (see F. Brooks' Mythical Man Month) and unfortunately we have to deal with it.

sigh

I will reduce the length of these fields from 255 to 150 for no other reason than that it is 10 times the length of a dotted-decimal IPv4 address and sufficient to accomodate a reverse DNS address twice ($2 \times 73 = 146$).

```
sessions.session_key: varchar(255) => varchar(172)
```

This field stores a session key, currently constructed using md5() which yields a string with 32 (lowercase) hexadecimal characters. In the future a different digest could be used to provide a session_key, e.g. SHA-1 (40 hexdigits) or SHA-512 (128 hexdigits). Another option would be to use a UUID: 128 bits represented in 32 hexdigits in the form xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx (string of 36 bytes). Alternatively, the SHA-512 could be encoded in base64 yielding a string of $512 / 6 = 86$ bytes. In this context, a field of size 255 seems a little over the top, not to mention problematic with 4-byte UTF-8 characters combined with the infamous MySQL / InnoDB-limit of 767 bytes for keyfields. I guess I will settle for a field size of 172 characters which is not too much for InnoDB keys + utf8mb4 and exactly enough to store a 1024 bit number in base64.

```
bool/function update_core 2012041900(&$output) [line 1874]
```

Function Parameters:

- *object* **&\$output** collects the html output

perform actual update to version 2012041900

Changes between 2011093000 and 2012041900:

- addition of the ckeditor-option in the site configuration table

bool function update_core_2013071100(&\$output) [*line 1979*]

Function Parameters:

- *object* **&\$output** collects the html output

perform actual update to version 2013071100

Changes between 2012041900 and 2013071100:

- addition of a new core table 'tokens'

bool function update_core_2014111700(&\$output) [*line 2017*]

Function Parameters:

- *object* **&\$output** collects the html output

perform actual update to version 2014111700

Changes between 2013071100 and 2014111700:

- addition of extra field in modules table
- addition of new site config parameter 'resize_dimension'
- addition of editor option 'ckeditor3' in site configuration

bool function update_core_2016062900(&\$output) [*line 2167*]

Function Parameters:

- *object* **&\$output** collects the html output

perform actual update to version 2016062900

Changes between 2013071100 and 2016062900:

- addition of new site config parameter 'cron_interval'
- addition of new site config parameter 'cron_next'
- addition of new site config parameter 'favicon'
- removal of foreign key constraint in 'nodes' table

bool function update_core_version(&\$output, \$version) [*line 955*]

Function Parameters:

- *object* **&\$output** collects the html output
- *int* **\$version** the new version number to store in config table

record the specified version number in the config table AND in \$CFG->version

This utility routine records the new version number in the config table and also adjusts the version number already in core (in \$CFG->version).

bool function update_create_table(\$tabledef) [*line 1096*]

Function Parameters:

- *array* **\$tabledef** contains the definition of a single table

create table in database from an individual tabledef

- **Uses \$DB**

bool function update_create_tables(\$filename) [*line 1072*]

Function Parameters:

- *string* **\$filename** contains the table definitions

create tables in database via include()'ing a file with tabledefs

void function update_language(&\$output, \$language_key, \$lanuage_key) [*line 406*]

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **\$language_key** primary key for language record in database AND name of the /program/languages subdirectory

- **\$language_key**

update a language in the database

this routine tries to update the information in the database with the information in the language manifest of the selected language \$language_key. The event is logged via logger().

Note that an upgrade of a language is not at all interesting because there is nothing to do except to update the data in the database with that from the manifest. However, we still do it this way in order for the user to grow accustomed to it so we can complicate this routine in the future without the user having to learn new tricks.

void function update_module(&\$output, \$module_key) [line 587]

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **\$module_key** unique secondary key for module record in modules table in database

call the module-specific upgrade routine

this routine tries to execute the correct upgrade script/function for module \$module_key. If all goes well, a success message is written to \$output (and the update is performed), otherwise an error message is written to \$output Either way the event is logged via logger().

Note that we take care not to load spurious files and execute non-existing functions. However, at some point we do have to have some trust in the file system...

bool function update_remove_obsolete_files(&\$output) [line 1834]

Function Parameters:

- *object* **&\$output** collects output

attempt to remove or at least flag obsolete files

this routine can grow bigger on every update when perhaps more files are obsoleted. We always check all files (even the older ones) because the user might not have removed them yet. If we can delete the files, we do so. If not, we log it and also show a message to the user via \$output.

void function update_show_overview(&\$output) [*line 166*]

Function Parameters:

- *object* **&\$output** collects the html output

display an introductory text for update + status overview

array function update_status_anchor([\$task = NULL], [\$key = NULL], [\$anchor = NULL]) [*line 930*]

Function Parameters:

- *string* **\$task** which update task do we need to do?
- *string||null* **\$key** which module/theme/etc. (NULL for core)
- *string* **\$anchor** text to show in link

return an anchor tag with link to the specific update function

This utility routine returns a ready to user HTML anchor tag.

void function update_status_table_close(&\$output) [*line 1012*]

Function Parameters:

- *object* **&\$output** collects the html output

close the status overview HTML-table we opened before

this is the companion routine for [update_status_table_open\(\)](#); it closes the open HTML-table

void function update_status_table_open(&\$output, [\$title = ""]) [*line 986*]

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **\$title** is the header of the first column

open a status overview HTML-table including column headers

this routine opens an HTML-table in preparation for a status overview of the system or a subsystem (languages, modules, themes). The optional title is used as the header of the first column.

The width of the first column is 25% and the remaining 5 columns area 15% each which creates an orderly display of name, internal version, external version, releasedate, release and status.

void function update_theme(&\$output, \$theme_key) [*line 802*]

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **\$theme_key** unique secondary key for theme record in themes table in database

call the theme-specific upgrade routine

this routine tries to execute the correct upgrade script/function for theme \$theme_id. If all goes well, a success message is written to \$output (and the update is performed), otherwise an error message is written to \$output Either way the event is logged via logger().

Note that we take care not to load spurious files and execute non-existing functions. However, at some point we do have to have some trust in the file system...

useraccount.class.php

/program/lib/useraccount.class.php - taking care of useraccounts

This file defines a class for dealing with users. Also, the global job permission constants and access control constants are defined. This file is always included, even when a visitor is anonymous (ie. not logged in).

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: useraccount.class.php,v 1.14 2016/05/18 15:42:43 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
ACL_ROLE_GURU = -1 [line 102]
ACL_ROLE_INTRANET_ACCESS = 1 [line 103]
ACL_ROLE_NONE = 0 [line 101]
ACL_ROLE_PAGEMANAGER_AREAMASTER = ACL_ROLE_PAGEMANAGER_SECTIONMASTER|
PERMISSION_AREA_DROP_PAGE|
PERMISSION_AREA_ADD_PAGE|
PERMISSION_AREA_DROP_SECTION|
PERMISSION_AREA_ADD_SECTION|
PERMISSION_AREA_EDIT_AREA [line 115]
ACL_ROLE_PAGEMANAGER_CONTENTMASTER = PERMISSION_NODE_EDIT_CONTENT [line 104]
ACL_ROLE_PAGEMANAGER_PAGEMASTER = ACL_ROLE_PAGEMANAGER_CONTENTMASTER|
PERMISSION_NODE_DROP_CONTENT|
PERMISSION_NODE_ADD_CONTENT|
PERMISSION_NODE_EDIT_PAGE [line 105]
ACL_ROLE_PAGEMANAGER_SECTIONMASTER = ACL_ROLE_PAGEMANAGER_PAGEMASTER|
PERMISSION_NODE_DROP_PAGE|
PERMISSION_NODE_ADD_PAGE|
PERMISSION_NODE_DROP_SECTION|
PERMISSION_NODE_ADD_SECTION|
PERMISSION_NODE_EDIT_SECTION [line 109]
ACL_ROLE_PAGEMANAGER_SITEMASTER = ACL_ROLE_PAGEMANAGER_AREAMASTER|
PERMISSION_SITE_DROP_AREA|
PERMISSION_SITE_ADD_AREA|
PERMISSION_SITE_EDIT_SITE [line 121]
JOB_PERMISSION_ACCOUNTMANAGER = 16 [line 47]
```

This (dangerous) permission allows access to add/edit/delete users and groups (including escalate privileges)

```
JOB_PERMISSION_BACKUPTOOL = 256 [line 59]
```

This allows the user to download a backup of the database

JOB_PERMISSION_CONFIGURATIONMANAGER = 32 *[line 50]*

This permission allows the user to access the configuration manager and change the site configuration

JOB_PERMISSION_FILEMANAGER = 4 *[line 41]*

This permission allows the user to access the file manager and upload/delete files in selected places

JOB_PERMISSION_GURU = -1 *[line 32]*

Guru permissions = all permission bits are set, even the unused ones

JOB_PERMISSION_LOGVIEW = 512 *[line 62]*

This allows the user to view the contents of the log table

JOB_PERMISSION_MASK = JOB_PERMISSION_NEXT_AVAILABLE_VALUE-1 *[line 77]*

This mask can be used to isolate only the 'official' permissions from an integer value

JOB_PERMISSION_MODULEMANAGER = 8 *[line 44]*

This permission allows the user to access the module manager and configure modules

JOB_PERMISSION_NEXT_AVAILABLE_VALUE = 4096 *[line 74]*

NOTE: This quasi-permission should always be defined to be the highest permission 1

JOB_PERMISSION_PAGEMANAGER = 2 *[line 38]*

This permission allows the user to access the page manager and add/edit/delete nodes according to the user's ACLs

JOB_PERMISSION_SESSIONTOOL = 2048 *[line 68]*

This allows the user to forcefully remove sessions [2016-05-18]

JOB_PERMISSION_STARTCENTER = 1 *[line 35]*

This permission is required for every user that is to logon to admin.php

JOB_PERMISSION_STATISTICS = 64 *[line 53]*

This permissions allows the user to access the site statistics

JOB_PERMISSION_TOOLS =

JOB_PERMISSION_TRANSLATETOOL|JOB_PERMISSION_BACKUPTOOL|JOB_PERMISSION_LOGVIEW|JOB_PERMISSION_UPDATE|JOB_PERMISSION_SESSIONTOOL *[line 71]*

combine the permssions for the tools in a single bit mask for convenient testing

JOB_PERMISSION_TRANSLATETOOL = 128 *[line 56]*

This allows the user to translate the program, by modifying existing translations or adding new languages

JOB_PERMISSION_UPDATE = 1024 *[line 65]*

This allows the user to perform a system upgrade (see [also version check\(\)](#) and [main_admin\(\)](#))

PERMISSION_AREA_ADD_PAGE = 1024 *[line 92]*

PERMISSION_AREA_ADD_SECTION = 4096 *[line 94]*

PERMISSION_AREA_DROP_PAGE = 512 *[line 91]*

PERMISSION_AREA_DROP_SECTION = 2048 *[line 93]*

PERMISSION_AREA_EDIT_AREA = 8192 *[line 95]*

PERMISSION_NODE_ADD_CONTENT = 4 *[line 82]*

PERMISSION_NODE_ADD_PAGE = 32 *[line 86]*

PERMISSION_NODE_ADD_SECTION = 128 *[line 88]*

PERMISSION_NODE_DROP_CONTENT = 2 *[line 81]*

PERMISSION_NODE_DROP_PAGE = 16 *[line 85]*

PERMISSION_NODE_DROP_SECTION = 64 *[line 87]*

PERMISSION_NODE_EDIT_CONTENT = 1 *[line 79]*

PERMISSION_NODE_EDIT_PAGE = 8 *[line 83]*

PERMISSION_NODE_EDIT_SECTION = 256 *[line 89]*

PERMISSION_SITE_ADD_AREA = 32768 *[line 98]*

PERMISSION_SITE_DROP_AREA = 16384 *[line 97]*

PERMISSION_SITE_EDIT_SITE = 65536 *[line 99]*

usermanager.class.php

/program/lib/usermanager.class.php - taking care of user management

This file defines a class for dealing with users.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: usermanager.class.php,v 1.22 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

GROUP_SELECT_ALL_USERS = -1 [*line 30*]

this value is used to select all users rather than users from a specific group

GROUP_SELECT_NO_GROUP = 0 [*line 33*]

this value is used to select the users that are not associated with any group

utf8lib.php

/program/lib/utf8lib.php - utility-routines for UTF-8

This file deals with the idiosyncrasies of UTF-8.

Reference: The Unicode Consortium, The Unicode Standard, Version 6.0.0, (Mountain View, CA: The Unicode Consortium, 2011, ISBN 978-1-936213-01-6) <<http://www.unicode.org/versions/Unicode6.0.0>>, chapter 3 (Conformance), page 94

Summary of the way valid code points are stored in 1 to 4 byte sequences.

bits	code-points	1st byte	2nd byte	3rd byte	4th byte
7	0000 0000 0000 0000	0xxx xxxx	0xxx.xxxx		
11	0000 0000 0000 0yyy yyxx xxxx	110y.yyyy	10xx.xxxx		
16	0000 0000 zzzz yyyy yyxx xxxx	1110.zzzz	10yy.yyyy	10xx.xxxx	
21	000w wwzz zzzz yyyy yyxx xxxx	1111.0www	10zz.zzzz	10yy.yyyy	10xx.xxxx

bits	byte 1	byte 2	byte 3	byte 4	comments
7	00 - 7F				U+0000 - U+007F
	80 - BF				--> ill-formed
	C0 - C1	80 - BF			--> overlong 2-byte
11	C2 - DF	80 - BF			U+0080 - U+07FF
	E0	80 - 9F*	80 - BF		--> overlong 3-byte
16	E0	A0 - BF*	80 - BF		U+0800 - U+0FFF
16	E1 - EC	80 - BF	80 - BF		U+1000 - U+CFFF
16	ED	80 - 9F*	80 - BF		U+D000 - U+D7FF
	ED	A0 - BF*	80 - BF		--> surrogates (U+D800 - U+DFFF)
16	EE - EF	80 - BF	80 - BF		U+E000 - U+FFFF
	F0	80 - 8F*	80 - BF	80 - BF	--> overlong 4-byte
21	F0	90 - BF*	80 - BF	80 - BF	U+10000 - U+3FFFF
21	F1 - F3	80 - BF	80 - BF	80 - BF	U+40000 - U+FFFFFF
21	F4	80 - 8F*	80 - BF	80 - BF	U+100000 - U+10FFFF
	F4	90 - BF*	80 - BF	80 - BF	--> invalid planes 11 - 13
	F5 - F7	80 - BF	80 - BF	80 - BF	--> invalid planes 14 - 1F
	F8 - FB				--> invalid 5-byte sequence
	FC - FD				--> invalid 6-byte sequence
	FE - FF				--> disallowed (BOM)

Note: Non-standard continuation ranges are marked with * (only for byte 2)

- **Package** wascore

- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: utf8lib.php,v 1.8 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

USE_MBSTRING = TRUE *[line 72]*

int function utf8_strcasecmp(\$utf8str1, \$utf8str2) [line 233]

Function Parameters:

- *string* **\$utf8str1** first string
- *string* **\$utf8str2** second string

compare two UTF8 strings in a case-INsensitive way

This compares two UTF-8 strings caseINsensitive. We do this by comparing the lowercase variant of the strings with the regular strcmp. We use lowercasing because that translation table is 'better' than the upper case one.

Note that this is a quick and dirty approach: we simply use the multibyte strings as-is rather than extracting the actual code points and applying some sort of collation because that would make it much more complicated.

int function utf8_strlen(\$utf8str) [line 164]

Function Parameters:

- *string* **\$utf8str** a valid UTF-8 string to examine

calculate the number of code points encoded in an UTF-8 string

This routine uses a trick to calculate the number of code points available in string \$str. By first converting the UTF-8 string to ISO-8859-1 all multi-byte code points (ie. all non-ASCII) are converted to a single byte: the correct ISO-8859-1 character where possible and a '?' for characters not available in the ISO-8859-1 repertoire. Note that utf8_decode() does NOT work very well on ill-formed UTF-8 strings, e.g. it happily interprets codes in invalid planes and translates truncated sequences to chr(0). The good news is that it works for valid UTF-8.

string function utf8_strtoascii(\$utf8str) [line 320]

Function Parameters:

- *string* **\$utf8str** input string possibly with letters with diacriticals etc.

map some UTF-8 characters to comparable ASCII strings

this maps a lot of UTF-8 characters to more or less comparable ASCII strings, e.g. an A-acute is mapped to 'A', etc. Handy when trying to construct readable filenames from letters with diacritics, eg. e-acute 'l' e-grave 'v' 'e' (French for pupil) maps to 'e' 'l' 'e' 'v' 'e' rather than 'l' 'v' 'e' if the diacriticals would simply be 'eaten'.

Note that UTF-8 characters that are NOT mapped to ASCII are retained, i.e. the result is not plain ASCII but an UTF-8 string with as much characters mapped to ASCII as possible.

- **Uses \$UTF8_ASCII**

string function utf8_strtolower(\$utf8str) [line 186]

Function Parameters:

- *string* **\$utf8str** a valid UTF-8 string to examine

fold a UTF-8 string to lower case

this routine tries to use the multibyte routine for folding, but if it is not available we fall back to our own translation table \$UTF8_UPPER_LOWER which is derived straight from the Unicode Character Database.

Informal benchmarking yielded no significant speed difference between mb_strtolower() and strtolower(). However, our table needs memory to store the 1100+ pairs and for mb_strtolower() this is already taken care of. OTOH: depending on the version of mb_string, our table (UCD 6.0.0 February 2011) may be more complete and up to date. Oh well.

string function utf8_strtoupper(\$utf8str) [line 207]

Function Parameters:

- *string* **\$utf8str** a valid UTF-8 string to examine

fold a UTF-8 string to upper case (sort of)

this routine tries to use the multibyte routine for folding, but if it is not available we fall back to our own translation table \$UTF8_LOWER_UPPER which is derived from the Unicode Character Database. Note that this table has some quirks because the underlying table is injective.

string function utf8_substr(\$utf8str, \$start, [\$length = NULL]) [*line 253*]

Function Parameters:

- *string* **\$utf8str** a valid UTF-8 string to examine
- *int* **\$start** an offset expressed in characters (not bytes)
- *int* **\$length** the length of the string to return (also expressed in characters)

return part of a UTF-8 string

this routine returns a valid UTF-8 substring from \$utf8str based on the \$start and \$length parameters in a way comparable to substr(). If available, we use the mb_substr() replacement, otherwise we perform the requested actions ourselves.

If we do it ourselves, we prefix variable names with 'c_' for characters and 'b_' for bytes. A UTF-8 character takes up to 4 bytes.

bool function utf8_validate(\$str) [*line 109*]

Function Parameters:

- *string* **\$str** the string to check

check an arbitrary string for UTF-8 conformity

return TRUE if string is valid UTF-8, FALSE otherwise.

the regular expression appears to crash PHP/Apache due to memory exhaustion when used on long(ish) strings. Therefore we validate only the short(ish) strings via a RE, while using the 22% slower algorithm for the longer ones. The pivot point of 4096 is an empirical value. It might mean trouble in the future, with different servers or OS's. We will cross that bridge when we get there... If you want to be on the safe side you could skip the RE-routine completely by lowering the pivot point to below 0.

Here is a broken down version of the RE:

```
$pattern = '/^([\x00-\x7F]' . // ASCII (including ctrl-chars)
'|[\xC2-\xDF][\x80-\xBF]' . // non-overlong 2-byte
'|[\xE0[\xA0-\xBF][\x80-\xBF]' . // 3-byte excluding overlongs
'|[\xE1-\xEC\xEE\xEF][\x80-\xBF]{2}' . // 3-byte (plain)
```

```
'|\\xED[\\x80-\\x9F][\\x80-\\xBF]'.          // 3-byte excluding surrogates
'|\\xF0[\\x90-\\xBF][\\x80-\\xBF]{2}'.        // 4-byte excluding overlongs
'|[\\xF1-\\xF3][\\x80-\\xBF]{3}'.            // 4-byte planes 4-15
'|\\xF4[\\x80-\\x8F][\\x80-\\xBF]{2})*$/';    // 4-byte plane 16
```

- Used by [ThemeRuta::ruta_get_background\(\)](#)

waslib.php

/program/lib/waslib.php - core functions

This file provides various utility routines.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: waslib.php,v 1.33 2016/05/26 13:22:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
CAPACITY_CHAIR = 8 [line 38]
CAPACITY_CUSTOM1 = 11 [line 41]
CAPACITY_CUSTOM2 = 12 [line 42]
CAPACITY_CUSTOM3 = 13 [line 43]
CAPACITY_CUSTOM4 = 14 [line 44]
CAPACITY_CUSTOM5 = 15 [line 45]
CAPACITY_CUSTOM6 = 16 [line 46]
CAPACITY_CUSTOM7 = 17 [line 47]
CAPACITY_CUSTOM8 = 18 [line 48]
CAPACITY_CUSTOM9 = 19 [line 49]
CAPACITY_EDITOR = 9 [line 39]
CAPACITY_MEMBER = 4 [line 34]
CAPACITY_NEXT_AVAILABLE = 20 [line 50]
CAPACITY_NONE = 0 [line 30]
```

The constants **CAPACITY_*** are used for group memberships (see [accountmanagerlib.php](#)).

```
CAPACITY_PRINCIPAL = 3 [line 33]
CAPACITY_PROJECTLEAD = 5 [line 35]
CAPACITY_PUBLISHER = 10 [line 40]
CAPACITY_PUPIL = 1 [line 31]
CAPACITY_SECRETARY = 7 [line 37]
CAPACITY_TEACHER = 2 [line 32]
CAPACITY_TREASURER = 6 [line 36]
QUASI_RANDOM_DIGITS = 10 [line 253]
QUASI_RANDOM_DIGITS_UPPER = 36 [line 255]
QUASI_RANDOM_DIGITS_UPPER_LOWER = 62 [line 256]
QUASI_RANDOM_HEXDIGITS = 16 [line 254]
string function appropriate_legal_notices([$text_only = FALSE], [$m = ""]) [line 1794]
```

Function Parameters:

- *bool* **\$text_only** if TRUE we return a text-only link, otherwise a clickable image
- *string* **\$m** margin to improve readability of generated code

construct a link to appropriate legal notices as per AGPLv3 section 5

This routine constructs ready-to-use HTML-code for a link to the Appropriate Legal Notices, which are to be found in /program/about.html. Depending on the highvisibility flag we either generate a text-based link or a clickabel image.

The actual text / image to use depends on the global constant WAS_ORIGINAL. This constant is defined in /program/version.php and it should be TRUE for the original version of Website@School and FALSE for modified versions.

In the former case the anchor looks like 'Powered by Website@School', in the latter case it will look like 'Based on Website@School', which is in line with the requirements from the license agreement for Website@School, see /program/license.html.

IMPORTANT NOTE

Please respect the license agreement and change the definition of WAS_ORIGINAL to FALSE if you modify this program (see /program/version.php). You also should change the file '/program/about.html' and add a 'prominent notice' of your modifications.

Note: a comparable routine can be found in [install.php](#).

array function calculate_uri_shortcuts(\$www, \$progwww) [line 189]

Function Parameters:

- *string* **\$www** the uri (scheme / authority / path) of the directory holding config.php
- *string* **\$progwww** the uri (scheme / authority / path) corresponding with the program directory

try to eliminate the scheme and authority from the two main uri's

This tries to get rid of the scheme and the authority in 'www' and 'progwww', If these two elements are the same, it becomes possible to use a shorter form of the uri when referencing files in 'progwww' from 'www'.

If the scheme and the authority of 'www' and 'progwww' are the same, the returned strings contain only the path elements. If scheme and authority differ, they contain the same as 'www' and 'progwww' respectively.

Examples: www = 'http://www.example.com/site' and progwww =
'http://www.example.com/site/program' yields www_short = " and wwwprog_short =

`'/program'.`

`www = 'http://www.example.com'` and `progwww = 'http://common.example.com/program'` yields `www_short` identical to `www` and `progwww_short` identical to `progwww`.

The purpose is to be able to generate relative links, e.g. an image in `/program/graphics/foo.jpg` can be referred to like this

```
 or  
 rather than  

```

Note that the comparison in this routine is not very fancy, it can be easily fooled to consider scheme+authority to be different. However, since this routine is only used to compare two values from `config.php`, it's not likely to cause trouble.

array function `calc_user_related_acls($user_id)` [line 1429]

Function Parameters:

- *int* **\$user_id** the user we're looking at

calculate an array with acls related to user \$user_id via group memberships

this calculates the related acls for user `$user_id`. The results are returned as an array keyed by `acl_id`. It can contain 0 or more elements. The values of the array elements are groupname/capacity-pairs. This routine is referenced from both [useraccount.class.php](#) and [usermanager.class.php](#).

string function `capacity_name($capacity)` [line 1406]

Function Parameters:

- *int* **\$capacity** numeric code of capacity

translate a numeric capacity code to a readable name

this translates a capacity code into a readable name, e.g. as an item in a dropdown list when dealing with group memberships. The actual codes are defined as constants, e.g. `CAPACITY_NONE`.

mixed function `convert_to_type($type, $value)` [line 1489]

Function Parameters:

- *string* **\$type** new type for `$value`: b=bool, i=integer, s=string, etc.

- *string* **\$value** the value to convert to type **\$type**

convert a string to another type (bool, int, etc.)

- **TODO** perhaps change the possible values of **\$type** to full strings rather than 'cryptic' single letter codes. Furthermore: what do we do with invalid dates, times and date/times? For now it is a stub, returning **\$value** as-is. Oh well.

int function cron_send_queued_alerts([\$max_messages = 10]) [*line 996*]

Function Parameters:

- *int* **\$max_messages** do not send more than this number of messages

send pending messages/alerts

this goes through all the alert accounts to see if any messages need to be sent out by email. The strategy is as follows. First we collect a maximum of **\$max_messages** alerts in core (1 trip to the database) Then we iterate through that collection and for every alert we

1. construct and send an email message
2. update the record (reset the message buffer and message count) (+1 trip to the database)

Locking and unlocking would be even more expensive, especially when chances of race conditions are not so big. (An earlier version of this routine went to the database once for the list of all pending alerts and subsequently twice for each alert but eventually I considered that too expensive too).

Assuming that an UPDATE is more or less atomic, we hopefully can get away with an UPDATE with a where clause looking explicitly for the previous value of the message count. If a message was added after retrieving the alerts but before updating, the message count would be incremented (by the other process) which would prevent us from updating. The alert would be left unchanged but including the added message. Worst case: the receiver gets the same list of alerts again and again. I consider that a fair trade off, given the low probability of it happening. (Mmmm, famous last words...)

Bottom line, we don't do locking in this routine.

Note that we add a small reminder to the message buffer about us processing the alert and sending a message. However, we don't set the number of messages to 1 because otherwise that would be the signal to send this message the next time. We don't want sent a message

every `$cron_interval` minutes basically saying that we didn't do anything since the previous run. (Or is this a feature after all?)

Failures are logged, successes are logged as `WLOG_DEBUG`.

string function `friendly_bookmark($title, [$maxlen = 50], [$ext = '.html'])` [line 2034]

Function Parameters:

- *string* **\$title** input text
- *int* **\$maxlen** the maximum length of the result
- *int* **\$ext** the filename extension added to a non-empty result

construct an alphanumeric string from a (node) title yielding a readable bookmark filename

this strips everything from `$title` except alphanumerics. Runs of other characters are translated to a single underscore. Length of result is limited to a length of `$maxlen` bytes (default 50). This includes the length of the extension `$ext`.

Note that the `$title` is UTF-8 and may contain non-ASCII characters. This routine deals with that situation by first converting the UTF-8 string to ASCII as much as possible (e.g. convert 'e-aigu' to plain 'e') and subsequently converting all remaining non-letters/digits to a underscores.

Finally the result is stripped from leading/trailing underscores. If this yields a non-empty string, the extension `$ext` (default `'.html'`) is appended.

Note: this route works best with latin-like text; if `$title` is completely written in Chinese (or other UTF-8 characters without a corresponding ASCII replacement) we end up with a single underscore which is subsequently `trim()`'ed, yielding an empty string and no `$ext` added. I am not sure what to do about that.

Note: the extension is not checked for non-alphanumerics because this is the responsibility of the caller to provide a decent `$ext` if the default `'.html'` is not used.

- **Used by** [was_node_url\(\)](#)

array/bool function `get_area_records([$forced = FALSE])` [line 1361]

Function Parameters:

- *bool* **\$forced** if TRUE forces reread from database (resets the cache)

retrieve a list of all available area records keyed by area_id

this returns a list of area-records or FALSE if no areas are available The list is cached via a static variable so we don't have to go to the database more than once for this. Note that the returned array is keyed with area_id and is sorted by sort_order. Also note that this list may include areas for which the current user has no permissions whatsoever and also areas that are inactive.

mixed function get_cookie_string(\$name, [\$default_value = NULL]) [*line 558*]

Function Parameters:

- *string* **\$name** the name of the cookie
- *mixed* **\$default_value** the value to return if cookie was not found

return an (unquoted) string value specified in the cookie header or default value if none

This validates and magic_unquotes() the specified cookie and returns either the valid UTF8 value or the UTF-8 substitution. If the cookie is not set in _COOKIE, the default value is returned. It is the responsibility of the caller to provide a workable default value.

array function get_csrf_token() [*line 2278*]

get csrf token name and value

this retrieves the current csrf token name and token value from \$_SESSION. If one of those is not already set we simply dream up a new one. This routine ALWAYS returns an array with two elements

array function get_editor_names() [*line 2395*]

prepare a list of available editors

this routine returns a hardcoded list of available editors: we do not expect to be adding or removing editors to/from the CMS on a regular basis, even though CKEditor 3 was added in March 2012 and CKEditor 4 was added in November 2014.

It might be cleaner to base this list on the site configuration options in the config table: a picklist of available editors is available in the 'editor' parameter in the table 'config'. The actual implementation of editors is done in [dialog_get_widget_richtextinput\(\)](#) in [dialoglib.php](#).

Here we (re-)use the translations for the (short) editor option and (long) editor name from the site config dialogs, e.g. via a constructed key 'site_config_editor_{\$editor}_option'.

Note: This routine is used by both the User Manager and the MyPage module.

- **TODO** retrieve this list from 'config'-table?

mixed function get_friendly_parameter(\$name, [\$default_value = NULL], [\$force = FALSE]) [*line 2098*]

Function Parameters:

- *string* **\$name** the parameter we need to look for
- *mixed* **\$default_value** is returned if the parameter was not found
- *bool* **\$force** if TRUE forces the parsing to be redone

retrieve a named parameter from the friendly URL

This routine attempts to parse the PATH_INFO server variable and extract the parameters and values stored in the path components. (see also [was_node_url\(\)](#)).

Example: the URL

/was/index.php/35/photo/5/Picture_of_our_field_trip.html

is broken down as follows:

- /35 is the first non-empty parameter and also is completely numeric and hence interpreted as a node_id;
- /photo/5 is considered a key-value-pair with key=photo and value=5;
- /Picture_of_our_field_trip.html is the last component and is discarded

The static array which caches the results of the parsing will contain this: \$parameters = array('node' => 35, 'photo' => 5);

Note that all parameters are checked for valid UTF-8. If either key or value is NOT UTF-8, the pair is silently discarded. This prevents tricks with overlong sequences and other UTF-8 black magic.

Once the parsed friendly path is cached the parameter \$name is looked up. If found, the corresponding value is returned. If it is not found, \$default_value is returned.

The cache is rebuilt if \$force is TRUE (should never be necessary)

Note: the parameter 'node' is a special case: if it is specified it is the first parameter. This parameter otherwise is unnamed.

array/bool function get_module_records([\$forced = FALSE]) [*line 1383*]

Function Parameters:

- *bool* **\$forced** if TRUE forces reread from database (resets the cache)

retrieve a list of all available module records

this returns a list of active module-records or FALSE if none are available The list is cached via a static variable so we don't have to go to the database more than once for this. Note that the returned array is keyed with module_id.

string function get_page_address_url([\$m = ""]) [*line 2354*]

Function Parameters:

- *string* **\$m** left margin for increased readability

return the reconstructed URL in a single (indented) line

This constructs the URL (including the GET-parameters and PATH_INFO) of the current script. The current script is identified using the basename of the entry point which is available in the global constant WASENTRY.

This URL is returned as HTML so it can be displayed. It is NOT meant to be a clickable link, but as a documentation of the actual URL that was used. Note that this URL can be suppressed by an appropriate 'display:none' in the stylesheet, making it an item that only appears on a hardcopy (media="print") and not on screen.

If somehow the input is invalid UTF-8, we replace the offending strings with the unicode substitution character U+FFFD in UTF-8 encode form (ie. the three character string 0xEF 0xBF 0xBD).

Note that we do need magic_unquote() because we are dealing with the PHP-version of parameters in \$_GET[] and \$_SERVER[] which - unfortunately - have magic quotes. We cannot use the routines because we would miss the last part of the PATH_INFO (the dummy 'filename'). Also, we need to validate the code for UTF-8 validity; we still do not want a malicious user somehow abusing /%C0%AE%2E/ (overlong UTF-8 equivalent of /../) to traverse the directory tree.

Note that two variants of this routine used to live in class Theme and class AdminOutput. Wrappers remain, though.

mixed function get_parameter_int(\$name, [\$default_value = NULL]) [*line 514*]

Function Parameters:

- *string* **\$name** the name of the parameter to retrieve the value of
- *mixed* **\$default_value** the value to return if parameter was not specified

return an integer value specified in the page request or default value if none

this routine first checks the friendly url to see if the requested parameter is specified there. If it is, we will use it unless there is also a parameter in \$_GET that prevails. If the parameter is not specified at all, the \$default_value is returned. It is the responsibility of the caller to provide a workable default value.

Note that invalid UTF-8 is silently discarded.

mixed function get_parameter_string(\$name, [\$default_value = NULL]) [*line 538*]

Function Parameters:

- *string* **\$name** the name of the parameter to retrieve the value of
- *mixed* **\$default_value** the value to return if parameter was not specified

return an (unquoted) string value specified in the page request or default value if none

First check out the friendly url for the named parameter. If it exists, we use that, otherwise we have the \$default_value. After that the valid UTF-8 value may overwrite the value found in the friendly url (or the default value).

It is the responsibility of the caller to provide a workable default value.

Note that invalid UTF-8 is silently discarded.

bool/array function get_properties([\$tablename = 'config'], [\$where = '']) [*line 116*]

Function Parameters:

- *string* **\$tablename** the name of the table holding the properties
- *array|string* **\$where** which records do we need to select

retrieve typed properties (name-value-pairs) from a table

this retrieves the fields 'name', 'value' and 'type' from all records from \$tablename that satisfy the condition in \$where. The values, which are stored as strings in the database, are converted to their proper value type and stored in the resulting array, keyed by name. The following types are recognised:

- `b` = boolean
- `d` = date ('yyyy-mm-dd', handled like a string)
- `dt` = datetime ('yyyy-mm-dd hh:mm:ss', handled like a string)
- `f` = float
- `i` = integer
- `s` = string
- `t` = time ('hh:mm:ss', handled like a string)

Note that we currently do not validate these properties, the assumption is that the values are valid (or empty).

int|null function `get_requested_area()` [line 467]

get the number of the area the user requested or null if not specified

See discussion of [get_requested_node\(\)](#).

string|null function `get_requested_filename()` [line 489]

get the name of the requested file

See discussion of [get_requested_node\(\)](#). Files are served via `/file.php` via a comparable mechanism: either

`http://exemplum.eu/was/file.php/path/to/filename.ext`

OR

`http://exemplum.eu/was/file.php?file=/path/to/filename.ext`

This routine extracts the `'/path/to/filename.ext'` part.

Note that we require valid UTF-8. If the path is not UTF-8, we return NULL.

int|null function `get_requested_node()` [line 456]

get the number of the node the user requested or NULL if not specified

This routine exists because nodes and (to a lesser extent) areas are so central to the whole idea of WAS.

A specific node can be requested in two different ways, for example page 35 with an additional parameter 'photo' with value 7 is called either via

`http://exemplum.eu/was/index.php/was/index.php/35/photo/5/Picture_of_our_field_trip.html`

or

`http://exemplum.eu/was/index.php/was/index.php?node=35&photo=5`

The routine `get_parameter_int()` with a default value of NULL yields 35 in both cases.

A node can also be specified implicitly, e.g. via

`http://exemplum.eu/was/index.php/was/index.php/area/1`

or

`http://exemplum.eu/was/index.php/was/index.php?area=1`

which yields the default node for area 1, or simply

`http://exemplum.eu/was/index.php/was/index.php`

which yields the default node in the default area.

Important note: In previous versions of this routine (and [get_requested_area\(\)](#)) we also accepted constructs like

`http://exemplum.eu/was/index.php/was/index.php/35`

`http://exemplum.eu/was/index.php/was/index.php/1/35`

but this has a great disadvantage that the idea of an ever growing list of integers (or more general: positional parameters) is not very handy in the long run. For instance: how to convey that we want to see photo #5 on page #35? 'index.php/35/5'? How is that to be interpreted different from page 5 in area 35? I now think the better approach is to use key/value-pairs in the friendly url path, and also get rid of the unnamed int indicating 'area'. The latter wasn't really usefull anyway, because specifying a node IMPLIES an area and it could even cause trouble if a bookmarked area+node would be moved to another area: the bookmark would yield an error message rather than the node (in another area) or the default node in the bookmarked area. All in all this change makes this routine extremely simple: it almost another name for `get_parameter_int()`. It is still possible to specify both node AND area (although there is no need):

`http://exemplum.eu/was/index.php/was/index.php/35/area/1`

`http://exemplum.eu/was/index.php/was/index.php/node/35/area/1`

`http://exemplum.eu/was/index.php/was/index.php/area/1/node/35`

Note that this routine does not validate the requested node in any way other than making sure that IF it is specified, it is valid UTF-8 and it is an integer value. For all we know it might even be a negative value.

array function get_skin_names() [line 2416]

prepare a list of available skins

this routine returns a hardcoded list of available skins: we do not expect to be adding or removing skins to/from the CMS any time soon.

Note: This routine is used by both the User Manager and the MyPage module.

int function get_unique_number([\$increment = TRUE]) [line 1755]

Function Parameters:

- *bool \$increment* optional indicates whether the static counter must be incremented

a small utility routine that returns a unique integer

this generates a unique number (starting at 1). This number is guaranteed to be unique during this http-request (or at least until the static variable \$id overflows, but that takes a while). If the optional parameter \$increment is FALSE, the latest id returned is returned again.

array function get_user_groups(\$user_id) [line 1458]

Function Parameters:

- *int \$user_id* the user we're looking at

retrieve the records of the groups of which user `$user_id` is a member

- **Uses** `$DB`

string function `hmac($key, $message, [$raw = FALSE], [$hash = "sha1"])` [line 2251]

Function Parameters:

- *string* **\$key** (shared) secret key
- *string* **\$message**
- *bool* **\$raw** TRUE return binary hmac, FALSE hexadecimal
- *function* **\$hash** either sha1 (default) or md5

calculate hmac according to RFC2104 (February 1997)

Note: strings `$opad` and `$ipad` are created by simply copying `$key` The contents are not important because we overwrite the contents in the loop anyway.

int function `ini_get_int($variable)` [line 1590]

Function Parameters:

- *string* **\$variable** name of the variable to retrieve, e.g. 'upload_max_filesize'

return an integer (bytecount) value from PHP ini

bool function `is_expired($node_id, &$tree)` [line 1323]

Function Parameters:

- *int* **\$node_id**
- *array* **&\$tree** family tree

determine if any of the ancestors or \$node_id itself is already expired

This climbs the tree upward, starting at \$node_id, to see if any nodes are expired. If an expired node is detected, TRUE is returned. If none of the nodes are expired, then FALSE is returned.

Note that this routine looks strictly at the expiry property, it is very well possible that a node is under embargo, see [is_under_embargo\(\)](#).

Also note that this routine currently also tries to 'fix' the node database when a circular reference is detected. This doesn't really belong here, but for the time being it is convenient to have this auto-repair mechanism here. The node that is fixed is the section we are looking at after MAXIMUM_ITERATIONS tries, which is not necessarily the node we started with.

- **TODO** this function also 'repairs' circular references. This should move to a separate tree-repair function but for the time being it is "convenient" to have automatic repairs...
- **Uses** \$DB

bool function is_under_embargo(&\$tree, \$node_id) [line 1275]

Function Parameters:

- *array* &\$tree family tree
- *int* \$node_id

determine if any of the ancestors or \$node_id itself is under embargo

This climbs the tree upward, starting at \$node_id, to see if any nodes are under embargo. If an embargo'ed node is detected, TRUE is returned. If none of the nodes are under embargo, then FALSE is returned.

Note that this routine looks strictly at the embargo property, it is very well possible that a node is expired, see [is_expired\(\)](#).

Also note that this routine currently also tries to 'fix' the node database when a circular reference is detected. This doesn't really belong here, but for the time being it is convenient to have this auto-repair mechanism here. The node that is fixed is the section we are looking at after MAXIMUM_ITERATIONS tries, which is not necessarily the node we started with.

- **TODO** this function also 'repairs' circular references. This should move to a separate tree-repair function but for the time being it is "convenient" to have automatic repairs...
- **Uses \$DB**

string function javascript_alert(\$message) [*line 298*]

Function Parameters:

- *string* **\$message** message to display

message a message and generate a javascript alert()

bool function lock_record(\$id, &\$lockinfo, \$tablename, \$pkey, \$locked_by, \$locked_since, [\$force = FALSE]) [*line 712*]

Function Parameters:

- *int* **\$id** the primary key of the record to lock
- *array* **&\$lockinfo** returns information about the session that already locked this record
- *string* **\$tablename** the name of the table
- *string* **\$pkey** name of the field holding the serial (pkey)
- *string* **\$locked_by** name of the field to hold our session_id indicating we locked the record
- *string* **\$locked_since** name of the field holding the datetime when the lock was obtained
- *bool* **\$force** TRUE means we grab the current lock from our other session

put a (co-operative) lock on a record

this tries to set the co-operative) lock on the record with serial (pkey) \$id in table \$tablename by setting the \$locked_by field to our own session_id. This is the companion routine of [lock_release\(\)](#).

The mechanism of co-operative locking works as follows. Some tables (such as the 'nodes' table) have an int field, e.g. 'locked_by_session_id'. This field can either be NULL (indicating that the record is not locked) or hold the primary key of a session (indicating that the record is locked and also by which session).

Obtaining a lock boils down to updating the table and setting that field to the session_id. As

long as the underlying database system guarantees that execution of an UPDATE statement is not interrupted, we can use UPDATE as a 'Test-And-Set'-function. According to the docentation MySQL does this.

The procedure is as follows.

1. we try to set the locked_by-field to our session_id on the condition that the previous value of that field is NULL. If this succeeds, we have effectively locked the record.
2. If this fails, we retrieve the current value of the field to see which session has locked it. If this happens to be us, we had already locked the record before and we're done.
3. If another session_id holds the lock, we check for that session's existence. If it still exists, we're out of luck: we can't obtain the lock unless \$force is TRUE. In that case we simply overrule the current lock and make it ours, if and only if the existing lock was granted to our user_id.
4. If that other session does no longer exist, we try to replace that other session's session_id with our own session_id, once again using a single UPDATE (avoiding another race condition). If that succeeds we're done and we have the lock; if it failes we're also done but without lock.

If locking the record fails because the record is already locked by another session, this routine returns information about that other session in \$lockinfo. It is up to the caller to use this information or not.

Note. A record can stay locked if the webbrowser of the locking session has crashed. Eventually this will be resolved if the crashed session is removed from the sessions table. However, the user may have restarted her browser while the record was locked. From the new session it appears that the record is still locked. This may take a while. Mmmmm... The other option is to lock on a per-user basis rather than per-session basis. Mmmm... Should we ask the user to override the session if it happens to be the same user? Mmm. put it on the todo list. (A small improvement might be to call the garbage collection between step 2 and 3. Oh well).

- **TODO** do we need a 'force lock' option to forcefully take over spurious locks? Maybe guru can override?
- **TODO** perhaps we can save 1 trip to the database by checking for something like `UPDATE SET locked_by = $session_id WHERE (id = $id) AND ((locked_by IS NULL) OR (locked_by = $session_id))` but I don't know how many affected rows that would yield if we already had the lock and effectively nothing changes in the record. (Perhaps always update atime to force 1 affected row?)
- **Usedby** [lock_release_node\(\)](#)

- Used by [lock_record_node\(\)](#)

bool function lock_record_node(\$node_id, &\$lockinfo, [\$force = FALSE]) [line 629]

Function Parameters:

- *int* **\$node_id** the primary key of the node to lock
- *array* **&\$lockinfo** returns information about the session that already locked this record
- *bool* **\$force** TRUE means we grab the current lock from our other session

get record lock on a node

this is a wrapper around [lock_record\(\)](#) for locking nodes.

- Uses [lock_record\(\)](#)

bool function lock_release(\$id, \$tablename, \$pkey, \$locked_by, \$locked_since) [line 816]

Function Parameters:

- *int* **\$id** the primary key of the record to unlock
- *string* **\$tablename** the name of the table
- *string* **\$pkey** name of the field holding the serial (pkey)
- *string* **\$locked_by** name of the field holding the session_id of the session that locked the record
- *string* **\$locked_since** name of the field holding the datetime when the lock was obtained

unlock a record that was previously successfully locked

this removes the co-operative) lock on the record with serial (pkey) \$id in table \$tablename by setting the \$locked_by field to NULL. This is the companion routine of [lock_record\(\)](#).

bool function lock_release_node(\$node_id) [*line 642*]

Function Parameters:

- *int* **\$node_id** the primary key of the node record to unlock

release lock on a node

this is a wrapper around [lock_release\(\)](#) for unlocking nodes.

- Uses [lock_record\(\)](#)

bool function logger(\$message, [\$priority = WLOG_INFO], [\$user_id = ""]) [*line 366*]

Function Parameters:

- *string* **\$message** the message to write to the log
- *int* **\$priority** loglevel, see PHP-function syslog() for a list of predefined constants
- **\$user_id**

a simple function to log information to the database 'for future reference'

This adds a message to the table log_messages, including a time, the remote address and (of course) a message. See also the standard PHP-function syslog(). We use the existing symbolic constants for priority. Default value is WLOG_INFO.

Note that messages with a priority WLOG_DEBUG are only written to the log if the global parameter \$CFG->debug is TRUE. All other messages are simply logged, no further questions asked.

If the caller does not provide a user_id, this routine attempts to read the user_id from the global \$_SESSION array, i.e. we try to link events to a particular user if possible.

Note that with a field definition of varchar(150) there is room to store either an IPv4 address (max 15 bytes) or a full-blown IPv6 address (39-47 bytes, see RFC3989) or even twice a complete reverse DNS address (see update_core_2011092100()).

See also [task_logview\(\)](#) for a rant on the difference between LOG_DEBUG and LOG_INFO.

- **TODO** should we make this configurable and maybe log directly to syslog (with automatic logrotate) or do we want to keep this 'self-contained' (the webmaster can read the table, but not the machine's syslog)?
- **Usedby** [login_send_bypass\(\)](#)
- **Usedby** [login_send_laissez_passer\(\)](#)
- **Uses** \$CFG

string function magic_unquote(\$value) [*line 83*]

Function Parameters:

- *string* **\$value** a string value that is conditionally unescaped

this circumvents the 'magic' in magic_quotes_gpc() by conditionally stripping slashes

Magic quotes are a royal pain for portability. If magic quotes are enabled, this function reverses the effect. There are three PHP-parameters in php.ini affecting the magic:

- the directive 'magic_quotes_runtime'
- the directive 'magic_quotes_gpc'
- the directive 'magic_quotes_sybase'

This routine deals with undoing the effect of the latter two. The effect of magic_quotes_runtime can be undone via set_magic_quotes_runtime(0). This is done once at program start (See [initialise\(\)](#) in [init.php](#)).

This routine should be used to unquote strings from \$_GET[], \$_POST[] and \$_COOKIE whenever they are needed.

Important note: because third party subsystems may deal with magic quotes on their own, it is a Bad Idea[tm] to globally replace the contents of \$_GET[], \$_POST[] and \$_COOKIE with the unescaped values once at program start. Any subsystem would be confused if magic_quotes_gpc() indicates that the magic is in effect whereas in reality the magic was already undone at program start. Yes, this yields a performance penalty, but this magic was a mess right from the start. Hopefully PHP6 will get rid of this magic for once and for all...

int function performance_get_queries() [*line 600*]

return the number of database queries that was executed

- **Uses \$DB**

double function performance_get_seconds() [*line 612*]

return the script execution time

- **TODO** maybe we should get rid of this \$PERFORMANCE object, because it doesn't do that much anyway

void function quasi_random_string(\$length, [\$candidates = 36]) [*line 282*]

Function Parameters:

- *int* **\$length** length of the string to generate
- *int* **\$candidates** number of candidate-characters to choose from

generate a string with quasi-random characters

This generates a string of \$length quasi-random characters. The optional parameter \$candidates determines which characters are eligible. Popular choices for \$candidates are:

- 10 (minimum): use only digits from 0,...,9
- 16: use digits 0,...,9 or letters A,...,F
- 36 (default): use digits 0,...,9 or letters A,...,Z
- 62: use digits 0,...,9 or letters A,...,Z or letters a,...,z

If \$candidates is smaller than 10, 10 is used, if \$candidates is greater than 62 62 is used.

Note that this is an ASCII-centric routine: we only use plain ASCII letters and digits and nothing of the 64000 other UNicode characters in the Basic Multilingual Plane. The reason is simple: 7-bit ASCII characters have the best chance of getting through communication channels unmangled so there.

void function queue_area_node_alert(\$areas, \$nodes, \$alert_message, [\$username = ""]) [*line 891*]

Function Parameters:

- *mixed* **\$areas** an array or a single int identifying the area(s) of interest
- *mixed* **\$nodes** an array or a single int identifying the node(s) of interest

- *string* **\$alert_message** the message to add to the buffer of qualifying alert accounts
- *string* **\$username** (optional) the name of the user that initiated the action

add a message to message queue of 0 or more alerts

this adds \$alert_message to the message buffers of 0 or more alert accounts The alerts that qualify to receive this addition via the alerts_areas_nodes table. The logic in that table is as follows:

- the area_id must match the area_id(s) (specified in \$areas) OR it must be 0 which acts as a wildcard for ALL areas
- the node_id must match the node_id(s) specified in \$nodes) OR it must be 0 which acts as a wildcard for ALL nodes

Also the account must be active and the flag for the area/node-combination must be TRUE.

As a rule this routine is called with a single area_id in \$areas and a collection of node_id's in \$nodes. The nodes follow the path up through the tree, in order to alert accounts that are only watching a section at a higher level.

Example: If user 'webmaster' adds new page, say 34, to subsection 8 in section 4 in area 1, you get something like this:

```
queue_area_node_alert(1,array(8,4,34),'node 34 added','webmaster');
```

The effect will be that all alerts with the following combinations of area A and node N have the message added to their buffers: A=0, N=1 - qualifies for all nodes in all areas A=1, N=0 - qualifies for all nodes in area 1 A=1, N=4 - qualifies for node 4 in area 1 A=1, N=8 - qualifies for node 8 in area 1

It is very well possible that no message is added at all if there is no alert watching the specified area and node (using wildcards or otherwise).

cron.php is to take care of eventually sending the queued messages.

Note that this routine adds a timestamp to the message and, if it is specified, the name of the user.

Also note that the messages are added to the buffer with the last message at the top, it means that the receiver will travel back in time reading the collection of messages. This is based on the assumption that the latest messages sometimes override a previous message and therefore should be read first.

- **Uses \$DB;**

string function quoted_printable(\$s, [\$textmode = TRUE], [\$newline = "\n"], [\$max_length = 76]) [*line* 1694]

Function Parameters:

- *string* **\$s** source string
- *bool* **\$textmode** TRUE means newlines count as hard line breaks, FALSE is binary data
- *string* **\$newline** native character indicating end of line
- *int* **\$max_length** indicates the limit for output lines (excluding the CRLF)

convert string \$s from native format to quoted printable (RFC2045)

this converts the input string \$s to quoted printable form as defined in RFC2045 (see <http://www.ietf.org/rfc/rfc2045.txt>). By default this routine assumes a line-oriented text input. This can be overruled by calling the routine with the parameter \$textmode set to FALSE: in that case the input is considered to be a binary string with no embedded newlines.

The routine assumes that the input lines are delimited with \$newline. By default this parameter is a LF (Linefeed) but it could be changed to another delimiter using the function parameter \$newline.

According to RFC2045 the resulting output lines should be no longer than 76 bytes, even though it is very well possible to use shorter lines. This can be done by setting the parameter \$max_length to the desired value. Note that this value is forced to be in the range 4,...,76.

The encoding is defined in section 6.7 of RFC2045 with these five rules.

- (1) General 8bit representation: any character may be represented as "=" followed by two uppercase hexadecimal digits.
- (2) Literal representation characters "!" to "~" but excluding the "=" may represent themselves.
- (3) White space Space " " and tab "\t" at the end of a line must use rule (1); in all other cases either rule (1) or (2) may be applied.
- (4) Line breaks The (hard) line breaks in the input must be represented using "\r\n" in the output.
- (5) Soft line breaks Output lines may not be longer than 76 bytes. This can be enforced by inserting a soft line break (the string "=\r\n") in the output. This soft line break will disappear once the encoded string is decoded.

The basic conversion algorithm is constructed using two important variables:

- an integer value (`$remaining`) indicating the number of bytes left in the current output line
- a boolean flag (`$next_is_newline`) indicating if the next input character is a `$newline`

The variable `$remaining` keeps track of situations where the current character (either as (1) General 8bit representation or (2) Literal representation) might not fit on the current line (eg. 2 bytes left requires an 8bit representation to be moved to the next output line). The flag `$next_is_newline` is used to make the best possible use of the available remaining space in the output, eg. if the current character is exactly as long as the remaining space, we can output that character on the current output line, because we are sure that it is the last character on the current output line so there cannot be a soft return next.

Note that spaces (ASCII 32) and tabs (ASCII 9) are treated differently depending on their position in the line. The rule is that both should be represented as `"=20"` or `"=09"` at the end of an input line and that it is allowed to use `" "` or `"\t"` when NOT at the end of an input line. In the latter case, the output line will allways end with a soft line break `"=\r\n"` which makes sure that there are not trailing spaces/tabs in the output line anyway.

Also note that the end of the input `$s` is also flagged via setting `$next_is_newline`. This is an optimisation which treats spaces and tabs at the end of the input as if they were at the end of an input line, ie. converting to `"=20"` or `"=09"`. This means that the output will never end with a space or a tab, even if the input does.

Note that in case of a binary conversion the input character(s) that might otherwise indicate a newline are to be considered as binary data. However, if the data is completely binary, it probably doesn't make sense to use Quoted-Printable in the first place (base64 would probably be a better choice).

Reference: see <http://www.ietf.org/rfc/rfc2045.txt>.

- **TODO** should we change the code to accomodate the canonical newline CRLF in the input?

nothing function `redirect_and_exit($url, [$message = ""])` [line 573]

Function Parameters:

- *string* `$url` the url to redirect to

- **\$message**

redirect to another url by sending an http header

string function `replace_crlf($multiline_string, [$replacement = ''])` [line 315]

Function Parameters:

- *string* **\$multiline_string** the multiline string to strip
- *string* **\$replacement** (optional) the string to replace newlines

unfold a possible multiline string

This removes all linefeeds and carriage returns from a string Typical use would be to strip a subject line in a mailmessage from newlines which might interfere with proper sending of mail headers.

string function `sanitise_filename($filename)` [line 1556]

Function Parameters:

- *string* **\$filename** the string to sanitise

sanitise a string to make it acceptable as a filename/directoryname

this routine analyses and maybe converts the input string as follows:

- all leading and trailing dots, spaces, dashes, underscores, backslashes and slashes are removed
- all embedded spaces, backslashes and slashes are converted to underscores
- only letters, digits, dots, dashes or underscores are retained
- all sequences of 2 or more underscores are replaced with a single underscore
- finally all 'forbidden' words (including empty string) get an underscore prefixed

Note that this sanitising only satisfies the basic rules for filenames; creating a new file with a sanitised name may still clash with an existing file or subdirectory.

Also note that a full pathname will yield something that looks like a simple filename without directories or drive letter: C:\Program Files\Apache Group\htpasswd becomes C_Program_Files_Apache_Group_htpasswd and /etc/passwd becomes etc_passwd. Also this routine makes a URL look like a filename: http://www.example.com becomes http_www.example.com.

Finally note that we don't even attempt to transliterate utf8-characters or any other characters

between 128 and 255; these are simply removed.

- **TODO** should we check for overlong UTF-8 encodings: C0 AF C0 AE C0 AE C0 AF equates to /../ or is that dealt with already by filtering on letters/digits and embedded dots/dashes/underscores?

string function short_datim(\$dt) [line 2299]

Function Parameters:

- *string* **\$dt** date/time to convert into short abbreviated form

construct an abbreviated date/time from a full date/time string

this converts the date/time 'yyyy-mm-dd hh:mm:ss' into either

- hh:mm (for today's dates)
- Ddd hh:mm (for the past week)
- yyyy-mm-dd (for the rest)

bool/long function string2time(\$timestring) [line 225]

Function Parameters:

- *string* **\$timestring** date/time in the form yyyy-mm-dd hh:mm:ss

convert a string representation of a date/time to a timestamp

this is a crude date/time parser. We collect digits and convert to integers. With the integers we fill an array with at least 6 integers, corresponding to year, month, day, hours, minutes and seconds. If there are less than six numbers in the source string the value 0 is used. for the remaining elements. Note that a number in this context is always a non-negative number because a dash (or minus) is considered a delimiter.

Note that valid date/time values are limited to how many seconds can be represented in a signed long integer, where 0 equates to 1970-01-01 00:00:00 (the Unix epoch). The upper limit for a 32-bit int is some date in 2038 (only 30 years from now).

string function t(\$phrase_key, [\$full_domain = "], [\$replace = "], [\$location_hint = "], [\$language = "]) [line 332]

Function Parameters:

- *string* **\$phrase_key** indicates the phrase that needs to be translated
- *string* **\$full_domain** (optional) indicates the text domain (perhaps with a prefix)
- *array* **\$replace** (optional) an assoc array with key-value-pairs to insert into the translation
- *string* **\$location_hint** (optional) hints at a directory location of language files
- *string* **\$language** (optional) target language

translation of phrases via a function with a very short name

This is only a wrapper function for `$LANGUAGE->get_phrase()`

- **Uses** `$LANGUAGE`

array function `tree_build($area_id, [$force = FALSE])` [line 1112]

Function Parameters:

- *int* **\$area_id** the area to make the tree for
- *bool* **\$force** if TRUE forces reread from database (resets the cache)

construct a tree of nodes in memory

this reads the nodes in the specified area from disk and constructs a tree via linked lists (sort of). If parameter `$force` is TRUE, the data is read from the database, otherwise a cached version is returned (if available).

Note that this routine also 'repairs' the tree when an orphan is detected. The orphan is automagically moved to the top of the area. Of course, it shouldn't happen, but if it does we are better off with a magically `_appearing_` orphan than a `_disappearing_` node.

A lot of operations in the page manager work with a tree of nodes in some way, e.g. walking the tree and displaying it or walking the tree and collecting the sections (but not the pages), etc.

The tree starts with a 'root' with key 0 (`$tree[0]`). This is the starting point of the tree. The nodes at the top level of an area are linked from this root node via the field 'first_child_id'. If

there are no nodes in the area, this field 'first_child_id' is zero. The linked list is constructed by using the node_id. All nodes in an area are collected in an array. This array is used to construct the linked lists.

Every node has a parent (via 'parent_id'), where the nodes at the top level have a parent_id of zero; this points to the 'root'. The nodes within a section or at the top level are linked forward via 'next_sibling_id' and backward via 'prev_sibling_id'. A zero indicates the end of the list. Children start with 'first_child_id'. A value of zero means: no children.

The complete node record from the database is also stored in the tree. This is used extensively throughout the pagemanager; it acts as a cache for all nodes in an area.

Note that we cache the node records per area. If two areas are involved, the cache doesn't work very well anymore. However, this doesn't happen very often; only in case of moving nodes from one area to another (and even then).

- **TODO** what if we need the trees of two different areas? should the static var here be an array, keyed by area_id?
- **TODO** repairing a node doesn't really belong here, in this routine. we really should have a separate 'database repair tool' for this purpose. someday we'll fix this....

bool function tree_visibility(\$subtree_id, &\$tree, [\$force_invisibility = FALSE]) [*line 1216*]

Function Parameters:

- *int* **\$subtree_id** the starting point for the tree walking
- *array* **&\$tree** pointer to the current tree
- *bool* **\$force_invisibility**

calculate the visibility of the nodes in the tree

this flags visible nodes as visible. Here 'visible' means that

- the node is not hidden, not expired and not under embargo
- the section has at least 1 visible node (page or section)

As a side effect, any subtree starting at a hidden/expired/embargo'ed section is completely set to invisible so we don't risk the chance to accidentally show a page from an invisible section. This routine walks through the tree recursively.

- **TODO** how about making all nodes under embargo visible when previewing a page or at least the path from the node to display?

bool function userdir_delete(\$path) [line 2193]

Function Parameters:

- *string \$path* the directory path relative to \$CFG->datadir, e.g. '/areas/exemplum' or '/users/acackl'

remove an 'empty' directory that used to contain (user)files

this removes the left-over files in the directory \$CFG->datadir.\$path and subsequently the directory itself. The allowable left-over files are those that are skipped in [userdir_is_empty\(\)](#). The (user) files we look at are those that are filtered out: - . and .. (directory housekeeping) - index.html of 0 bytes ('protects' directory from prying eyes) - symbolic links - thumbnails (filenames starting with THUMBNAIL_PREFIX) This filtering is the same as that in the file manager (see [filemanager.class.php](#)).

Note that any symbolic links are deleted too.

bool function userdir_is_empty(\$path) [line 2145]

Function Parameters:

- *string \$path* the directory path relative to \$CFG->datadir, e.g. '/areas/exemplum' or '/users/acackl'

determine whether a directory is empty (free from (user)files)

this scans the directory \$CFG->datadir.\$path to see if it is empty, i.e. does not contain any (user)files. Returns TRUE if empty, FALSE otherwise. The (user) files we look at are those that are not filtered out: - . and .. (directory housekeeping) - index.html of 0 bytes ('protects' directory from prying eyes) - symbolic links - thumbnails (filenames starting with THUMBNAIL_PREFIX) This filtering is the same as that in the file manager (see [filemanager.class.php](#)).

string function was_file_url(\$path, [\$fully_qualified = FALSE]) [line 1895]

Function Parameters:

- *string* **\$path** the name of the file including path
- *bool* **\$fully_qualified** if TRUE forces the URL to contain a scheme, authority etc., else use shortened form

construct a url that links to a file via /file.php

This constructs a URL that links to a file, either

/file.php/path/to/file.txt

or

/file.php?file=/path/to/file.txt

depending on the global setting for proxy-friendly urls.

Furthermore, if the flag **\$fully_qualified** is TRUE, we include scheme and authority in the resulting URL, ie.

<http://exemplum.eu/file.php/path/to/file.txt>

- Usedby [ThemeCornelia::cornelia_get_background_url\(\)](#)
- Usedby [ThemeRuta::ruta_get_background\(\)](#)

string function was_node_url([\$node = NULL], [\$parameters = NULL], [\$bookmark = ""], [\$preview = FALSE], [\$qualified = FALSE]) [line 1957]

Function Parameters:

- *array* **\$node** record straight from the database (or \$tree)
- *array|null* **\$parameters** additional parameters for the url (path components in friendly url mode)
- *string* **\$bookmark** the basis for a visual clue to identify the node (in friendly url mode only)
- *bool* **\$preview** if TRUE, the href is replaced with a bare '#' to obstruct navigation in preview mode
- *bool* **\$qualified** if TRUE use the scheme and authority, otherwise use the short(er) form without scheme/authority

construct a ready-to-use href which links to the node \$node via index.php

this routine creates a ready-to-use href that links to node \$node, taking these options into account:

- the href is replaced with a bare '#' if we area in preview mode
- the href is either fully qualified or abbreviated, depending on \$qualified
- the node_id is conveyed either as a proxy-friendly url or a simple parameter

?node=\$node_id

- if we use friendly url, \$node_id always comes first in the path (without the word 'node')
- if we use friendly url, \$bookmark is appended as the last item in the path
- the additional parameters (if any) are sandwiched between the node_id and the bookmark

This routine mainly deals with constructing a friendly url taking parameters into account in the form of path components. The node_id is conveyed as the first parameter and it has no associated name, ie. the url is shortened from '/was/index.php/node/35' to '/was/index.php/35'. All other parameters from the array \$parameters (if any) are added as pairs: '/key1/value1/key2/value2/key3/value3' etc. The last parameter added to this path is based on the \$bookmark or, if that is empty the node's title. The purpose of this parameter is to create a URL that looks like a descriptive filename, which makes it easier for the visitor to bookmark this page and still have a clue as to what the page is about. Otherwise this parameter is not used at all; the 'real' navigation information is in the node_id and the additional parameters. Example:

```
was_node_url($tree[35]['record'],array('photo'=>'5'),'Picture of our field trip')
```

yields the following URL (when friendly urls are used):

```
/was/index.php/35/photo/5/Picture_of_our_field_trip.html
```

or (when friendly urls are not used):

```
/was/index.php?node=35&photo=5
```

The interesting bits are the node_id (35) and the photo_id (5). The string alias filename 'Picture_of_our_field_trip.html' is merely a suggestion to the browser and is not used by W@S.

- Used by [Theme::node2anchor\(\)](#)
- Uses [friendly_bookmark\(\)](#)
- Uses \$CFG

string function was_url(\$url, [\$fully_qualified = FALSE]) [*line 1853*]

Function Parameters:

- *string* **\$url** the (possibly) relative URL to massage
- *bool* **\$fully_qualified** if TRUE forces the URL to contain a scheme, authority etc., else use shortened form

massage a possibly relative URL to make it more qualified

Here we perform some heuristics: if \$url looks like it is relative, we prepend the correct path (from \$CFG) to it.

Here a URL is considered relative when it does NOT start with a slash and it does NOT start with a scheme followed by '://'. Additionally, we make a distinction between a relative URL starting with 'program/' (which indicates a static file somewhere in the program directory) and

other relative URLs (which are assumed to start in the CMS Root Directory (the directory where index.php, admin.php & friends live)).

Note: according to RFC3986 a scheme must start with a letter and can contain only letters, digits, '+', '-' or '.'. Note: all string operations here are ASCII; no UTF-8 issues here.

If `$fully_qualified` is TRUE we always make a relative URL fully qualified.

If `$url` starts with a slash, we must assume that the caller means some file relative to the document root, or rather: relative to the top level directory embedded in `$CFG->www`. If we have `$url` starting with a slash AND `$full_qualified` is TRUE, we extract the scheme and authority from `$CFG->www` and prepend that to `$url`. This is a heuristic approach.

Example 1: `'program/styles/base.css'` becomes `'/program/styles/base.css'` OR `'http://exemplum.eu/program/styles/base.css'`

Example 2: `'file.php/areas/exemplum/logo.jpg'` becomes `'/file.php/areas/exemplum/logo.jpg'` OR `'http://exemplum.eu/file.php/areas/exemplum/logo.jpg'`

Example 3: `'/path/to/foo/bar/logo.jpg'` becomes `'/path/to/foo/bar/logo.jpg'` OR `'http://exemplum.eu/path/to/foo/bar/logo.jpg'`

- Usedby [ThemeCornelia::cornelia_get_background_url\(\)](#)
- Usedby [ThemeRuta::ruta_get_background\(\)](#)
- Usedby [Theme::add_stylesheet\(\)](#)

zip.class.php

/program/lib/zip.class.php - create simple ZIP-archives

This file implements class Zip which allows for creating ZIP-archives on the fly

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: zip.class.php,v 1.6 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

ZIP_TYPE_BUFFER = buffer *[line 32]*
ZIP_TYPE_FILE = file *[line 30]*
ZIP_TYPE_NONE = ' *[line 29]*
ZIP_TYPE_STREAM = stream *[line 31]*

main_admin.php

/program/main_admin.php - workhorse for site maintenance

This file deals with the administrator interface program for site maintenance. It is included and called from /admin.php.

The work is done in [main_admin\(\)](#).

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: main_admin.php,v 1.27 2016/03/25 15:53:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

JOB_ACCOUNTMANAGER = accountmanager *[line 55]*

This is used to dispatch the account manager (users and groups)

JOB_CONFIGURATIONMANAGER = configurationmanager *[line 58]*

This is used to dispatch the configuration manager

JOB_FILEBROWSER = filebrowser *[line 43]*

This is used to dispatch the file manager in file browser mode (used with CKEditor and FCKeditor)

JOB_FILEMANAGER = filemanager *[line 40]*

This is used to dispatch the file manager

JOB_FLASHBROWSER = flashbrowser *[line 49]*

This is used to dispatch the file manager in flash browser mode (used with CKEditor and FCKeditor)

JOB_IMAGEBROWSER = imagebrowser *[line 46]*

This is used to dispatch the file manager in image browser mode (used with CKEditor and FCKeditor)

JOB_MODULEMANAGER = modulemanager *[line 52]*

This is used to dispatch the module manager

JOB_PAGEMANAGER = pagemanager *[line 37]*

This is used to dispatch the page manager

JOB_STARTCENTER = start *[line 34]*

This is used to dispatch the startcenter job

JOB_STATISTICS = statistics *[line 61]*

This is used to dispatch the statistics

JOB_TOOLS = tools *[line 64]*

This is used to dispatch the tool manager

JOB_UPDATE = update *[line 67]*

This is used to dispatch the update manager

void function add_javascript_popup_function(&\$output, [\$m = ""]) [line 502]

Function Parameters:

- **&\$output**
- **\$m**

add javascript code that implements a popup to the header part of the page

void function add_javascript_select_url_function(&\$output, [\$m = ""]) [line 555]

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **\$m** left margin for increased readability

add javascript code that implements a url selection (used in integration with CKEditor/FCKeditor)

This adds a JavaScript-function to the currently generated output page which takes care of returning a URL from a file/image/flash browser to either the (older) FCKeditor or (newer)

CKEditor. Since both editors use the same filebrowsers we need to discriminate between FCKEditor and CKEditor. This is done by looking at the parameters: the CKEditor provides the number of an anonymous function in the parameter 'CKEditorFuncNum'. If this parameter is set we use the (integer) value for the callback to CKEditor. If it is not set we assume the old interface with FCKEditor.

Note that our actual file browser is supposed to remember this parameter, otherwise the file browser will assume FCKEditor after navigating to another page within the file browser. See also [filemanager.class.php](#). We remember the parameter via a session variable which is easier than propagating this number by adding it to every link within the file browser. However, we do update this function number every time the parameter 'CKEditorFuncNum' is specified, ie. on the first call to a file browser from CKEditor, ie. whenever the user starts browsing the server.

void/int function admin_continue_session() [line 418]

continue the session from the previous call OR exit

This tries to resume the session that was initiated before (when the user logged in successfully). If the session cannot be resumed, we logout the user, show the login screen and exit. In other words: this routine guarantees that a valid session exists if and when this routine returns. If the routine returns, it returns the user_id.

- **Uses** [dbsessionlib.php](#)
- **Uses** \$CFG;

void/int function admin_login(\$step) [line 388]

Function Parameters:

- *mixed* **\$step** the step in the login procedure

perform a step in the login procedure

This routine may not return at all. If it returns, the user is logged in successfully and the return value is the user_id (unique identification for the user, pkey in users table). The steps in the login procedure are defined in [loginlib.php](#).

- Uses [loginlib.php](#)
- Uses \$CFG

void function admin_logout_and_exit() [*line 366*]

logout the user and exit

This logs out the user (ie kills the session) If there is an error (ie, there was no session in the first place) indicated by was_logout() returning, we unconditionally show a login dialog and exit. So, this routine never returns.

- Uses [loginlib.php](#)
- Uses \$CFG

void function admin_show_login_and_exit([\$message = ""]) [*line 450*]

Function Parameters:

- \$message

show login dialog and exit

- Uses [loginlib.php](#)
- Uses \$CFG

string function get_current_skin() [*line 1728*]

determine the default skin to use

This routine determines which skin to use in AdminOutput. It always returns a valid skin (using 'base' in case of error).

- **Uses** \$_SESSION;
- **Uses** \$USER;

string function get_versioncheck_url() [*line 587*]

construct URL for version check against the project's website

this constructs the URL for checking the installed version against the current version on the project's website. The remote site will respond with a readable text and the user can decide to act on the information (or not). We don't want to force any upgrades etc.

void function job_start(&\$output) [*line 302*]

Function Parameters:

- *object* **&\$output** output collector

generate the start centre page

This is the handler for the start centre. This is the only handler that is NOT included from another file. Basically it shows a screen with hints on how to proceed with this program.

- **TODO** this routine is a stub

void function main_admin() [*line 105*]

main program for site maintenance

This is the main administrator program. First step is to deal with users logging in or out. If a user is not logged in, a login dialog is displayed. If a user is logged in but has no admin privileges, she is redirected to the public site (ie. index.php).

Once we have established that the user is an administrator, we setup an output collecting object and see what the user wants us to do by interpreting the parameter 'job'. If the user has access to the specified job, the corresponding code is included and the main routine of that handler is called. It is then the responsibility of that handler to further decide what needs to be done. After the handler returns, the collected output is sent to the user. This includes the main navigation (i.e. links to the various 'managers') and also the menu and the content generated by the handler.

If the user has no privilege to access a particular manager, an error message is displayed in both the message area and the content area. This makes it clear to the user that access is denied. Note that the inaccessible items are displayed in the main navigation via 'dimmed' (light-grey) links or black/white images. By showing these 'dimmed' links, the user will be aware that there is more than just what she is allowed to see. This is more transparent than suppressing items and keeping them secret.

- **TODO** should we cater for a special 'print' button + support for a special style sheet for media="print"?
- **Uses** \$USER;
- **Uses** \$LANGUAGE;
- **Uses** \$CFG;

void function non_admin_redirect_and_exit() [line 469]

tell non-admin-user access denied and exit

this routine shows a supersimple screen for an intranet user (ie. someone without admin privileges) who accidentally hit admin.php to go and look elsewhere by presenting a link to index.php or to admin.php?login=1.

- **Uses** \$CFG

main_cron.php

/program/main_cron.php - take care of recurring jobs

This file deals with executing cron jobs. It is included and called from /cron.php.

The work is done in [main_cron\(\)](#).

First we look to see if it is already time to do something. Criterion: the current time is \geq the time in `$CFG->cron_next`. If that is the case, we start with calculating the time of the next run by adding `$CFG->cron_interval` minutes to the current time.

Note: we calculate with minutes of 57 seconds just to have a 5% margin between the system cron clock and our own handling. It might just be enough if cron calls us every 15 minutes (900s) and we schedule the `_next_` run for 15 minutes (855s): this gives us 45s to deal with delays (otherwise we have to wait $2 \times 15 = 30$ minutes for the next run).

After scheduling the next run, we step through all modules and execute any cronjobs there in much the same way (scheduling the next run before executing the current one, etc.).

Note that this cronjob is called from /cron.php. This script could be added to /etc/crontab by adding the following line:

```
1 * * * * www /usr/bin/php /path/to/cron.php
```

This will let the PHP CLI interpreter run cron.php one minute after the hour, every hour. Note that the 'www' is the userid to run the script as. (The user running the script must have permissions to read all included PHP-files, including config.php).

Alternatively it is possible to periodically run a cronjob which executes wget which in turn retrieves the cron.php script via the web server, e.g.

```
31 * * * * mmonte /usr/bin/wget -q http://exemplum.eu/cron.php
```

to call cron.php every 31 minutes past the hour, every hour. You could suppress any output by adding `-q -O /dev/null`. See man wget for more information.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: main_cron.php,v 1.9 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **TODO** should we act differently based on how we are called (cli or web), eg send more queued alerts?

- **TODO** should we build extra access protection via passwords and/or IP-addresses?
- **License** [GNU AGPLv3+Additional Terms](#)

void function main_cron() [*line 73*]

main_file.php

/program/main_file.php - workhorse for serving files

This file deals with serving files It is included and called from /file.php.

The work is done in [main_file\(\)](#).

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: main_file.php,v 1.11 2016/06/02 16:28:12 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

void/bool function download_source(\$component) [*line 346*]

Function Parameters:

- *string* **\$component** either 'program' or 'manual' or 'languages'

construct a zipfile with the current source and stream it to the visitor

this routine streams a ZIP-archive to the visitor with either the current websiteatschool program code or the selected manual. This routine is necessary to comply with the provisions of the program license which basically says that the source code of the running program must be made available.

Note that it is not strictly necessary to also provide the manual, but this routine can do that nevertheless.

Note that we take special care not to download the (private) data directory \$CFG->datadir. Of course the datadirectory should live outside the document root and certainly outside the /program directory tree, but accidents will happen and we don't want to create a gaping security hole.

If there are severe errors (e.g. no manual is available for download or an invalid component was specified) the program exist immediately with a 404 not found error. Otherwise the ZIP-archive is streamed to the user. If all goes well, we return TRUE, if there were errors we immediately return TRUE (without finishing the task at hand other than a perhasp futile attempt to properly close the ZIP-archive). The last error message from the Zip is logged.

- Uses [download_source_tree\(\)](#)

bool function download_source_tree(&\$zip, \$path, \$vpath, &\$excludes) [line 478]

Function Parameters:

- *object* **&\$zip** Zip-archive
- *string* **\$path** physical directory to add to archive
- *string* **\$vpath** virtual pathname for this physical directory
- *array* **&\$excludes** array with 'forbidden' subdirectories

workhorse function to recursively add most of a tree to a ZIP-archive

this routine recursively adds the tree starting at \$path to the opened archive \$zip. If a directory is in the list of excluded directories in \$excludes it is skipped.

- Usedby [download_source_tree\(\)](#)
- Usedby [download_source\(\)](#)
- Uses [download_source_tree\(\)](#)

void function error_exit404([\$filename = ""]) [line 313]

Function Parameters:

- *string* **\$filename** the file we were looking for and could not find

exit with a 404 not found error

void function main_file() [line 99]

main program for serving files

this routine is called from /file.php.

This routine is responsible for serving files to the visitor. These files are stored in a (virtual) file hierarchy that looks like this.

```
/areas/areaname
  /another
  /stillmore
...
/users/username
  /another
  /stillmore
...
/groups/groupname
  /another
  /stillmore
...
/websiteatschool/program
  /manual
  /languages
```

This structure maps to the real file system as follows. The (virtual) directories /areas, /users and /groups correspond to the physical directories {`$CFG->datadir`}/areas, {`$CFG->datadir`}/users and {`$CFG->datadir`}/groups respectively. The subdirectories correspond to a (unique) area, user or group and serve as a file repository for that area, user or group.

The (virtual) top-level directory /websiteatschool is a special case. It is used to serve the currently running website program code and the user-defined translations of active languages.

Before any file is transmitted to the visitor the access privileges are checked. The following rules apply.

Access control for the /areas subdirectory

- an area must be active before any files are served
 - the visitor must have access to the private area if files are to be served
 - non-existing files yield a 404 Not Found error
 - non-existing areas also yield a 404 Not Found error
 - if the visitor has no access to the private area, also a 404 Not Found error is returned
- Access control for /users and /groups

- a user/group must be active before any files are served
 - non-existing users/groups yield 404 Not Found
 - non-existing files in existing directories also yield 404 Not Found
- Access control for /websiteatschool

- there is no limit on downloading the currently active program code or user-defined

translations of active languages

Note: The check on '..' in the requested filename would be inconclusive if the \$path is encoded in invalid UTF-8: the overlong sequence 2F C0 AE 2E 2F eventually yields 2F 2E 2E 2F or '/../'. Reference: RFC3629 section 10. However, we use the filename processed with get_requested_filename() which already checks for utf8 validity, which rules out the trick with overlong sequences.

bool/int function readfile_chunked(\$path) [*line 518*]

Function Parameters:

- *string* **\$path** fully qualified path of the file to send

send a file to the visitor's browser in chunks

This sends the file \$path to the browser in manageable chunks.

string function rfc1123date([\$t = 0]) [*line 202*]

Function Parameters:

- *int* **\$t** the date/time value to use, or 0 for current time

generate an RFC1123-compliant date/time stamp

This constructs a date/time stamp that is a fixed-length subset of RFC1123. This is the preferred format in HTTP (see RFC2616 section 3.3).

The format is as follows: rfc1123-date = wkday ", " SP date SP time SP "GMT"
date = 2DIGIT SP month SP 4DIGIT ; day month year (e.g., 02 Jun 1982)
time = 2DIGIT ":" 2DIGIT ":" 2DIGIT ; 00:00:00 - 23:59:59
wkday = "Mon" | "Tue" | "Wed" | "Thu" | "Fri" | "Sat" | "Sun"
month = "Jan" | "Feb" | "Mar" | "Apr" | "May" | "Jun" |
"Jul" | "Aug" | "Sep" | "Oct" | "Nov" | "Dec"

If \$timevalue is less or equal to zero, the current time is used, otherwies \$timevalue is interpreted as a standard unix timestamp.

bool/int function send_file_from_datadir(\$file, \$name, [\$mimetype = ""], [\$ttl = 86400], [\$download = FALSE]) [*line 251*]

Function Parameters:

- *string* **\$file** name of the file to send relative to \$CFG->datadir

- *string* **\$name** filename to suggest to the visitor/visitor's browser
- *string* **\$mimetype** the mime type of the file; if not specified we look it up
- *int* **\$ttl** time to live (aka maximum age) in seconds, 0 implies file is not cacheable
- *bool* **\$download** if TRUE we try to force a download

the designated file is sent to the visitor

This transmits the file `{ $CFG->datadir }$file` from the data directory to the visitor's browser, suggesting the name `$name`. The file is transmitted in chunks (see [readfile_chunked\(\)](#)).

Several different variations are possible.

- by specifying a Time To Live of 0 seconds, this routine tries hard to defeat any caching by proxies
- if the download flag is TRUE, this routine tries to prevent the visitor's browser to render the file in-line suggesting downloading instead
- Quirks
- There appears to be a problem with Internet Explorer and https:// and caching which requires a specific workaround. We simply check for 'https:' or 'http'.
- Adobe Acrobat Reader has a bad track record of infecting user's computers with malware when PDF's are rendered in-line. Therefore we force download for that kind of files by checking for extension 'pdf' or mediatype 'application/pdf' or 'application/x-pdf'.
- It is not easy to determine the exact mime type of files without resorting to a complex shadow-filesystem or a metadata table in the database. Therefore we rely on the information provided via `finfo`, `mime_content_type()` or `file(1)`. See [get_mimetype\(\)](#) for details.

- Uses [get_mimetype\(\)](#)
- Uses [get_mediatype\(\)](#)

main_index.php

/program/main_index.php - workhorse for visitor interface

This file deals with the visitor interface. It is included and called from /index.php.

The work is done in [main_index\(\)](#).

Parameters are passed like index.php?parm=val. Here is an overview of recognised global parameters that are handled here (and not in a module).

- **logout** - used to end a user's session (no need for a value, just the param is enough)
- **login=i** - step i of the login procedure
- **area=a** - indicates which area to access; if specified it should match the node, if that is specified
- **node=n** - indicates which node to access; if specified it should match the area, if that is specified
- **language=xx** - indicates the language to use; xx is a valid language code like 'en', 'de', 'fr' or 'nl'

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: main_index.php,v 1.15 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **TODO** add the performance results in a HTML-comment if not CFG->debug, in sight otherwise
- **License** [GNU AGPLv3+Additional Terms](#)

bool|array function calculate_area(\$requested_area, \$requested_node) [line 284]

Function Parameters:

- *int|null* **\$requested_area** the area the user specified or NULL if no area specifically requested
- *int|null* **\$requested_node** the node the user specified or NULL if no node specifically requested

try to retrieve a valid area record based on values of requested area and requested node

this determines which area to use. If the user specifies nothing (no area, no node), we

simply go for the default area or the first available area. If the user does specify an area and/or a node, we use that information to get to the area. Note that if the user specifies both area and node, the two should match. That is: you cannot specify a node from area X and also request area Y: that yields no results. If only a node is specified, the area is calculated from the area to which the node belongs.

We let the database do most of the work by constructing and executing an appropriate SQL-statement.

- **Uses \$DB**

bool/int function calculate_default_page(&\$tree, \$subtree_id) [line 377]

Function Parameters:

- *array* **&\$tree** a reference to the complete tree in the area of interest
- *int* **\$subtree_id** the place where we need to start looking (usually the first_child_id of the parent)

try to find a default page within a subtree of pages and sections

this walks the tree \$tree starting at \$subtree_id looking for a default page. We give it three tries. First we look for a default node in the section of which \$subtree_id is the first node. If we find a page, we're done, if we find a section we descend into that subtree. If there still is no default page, we go look for any page in the initial set of nodes. If that too doesn't yield a page, we descend into the subtrees. If THAT doesn't yield a page we give up and return FALSE, indicating no page to be found.

bool/int function calculate_node_id(&\$tree, \$area_id, \$requested_node) [line 344]

Function Parameters:

- *array* **&\$tree** a reference to the complete tree in area \$area_id
- *int* **\$area_id** the area where we are looking for a node
- *int/null* **\$requested_node** the node_id the user requested or NULL if none was specified

calculate and validate the node_id to display

this tries to determine a valid node to display based on the node the user requested and

the area that the user may or may not have requested.

Basic assumption is that the visitor has indeed view access to area \$area_id. This means that the user is allowed to see the nodes in this area that are not under embargo (and not expired). We do have a complete overview of all nodes in this area in the array \$tree. (See [tree_build\(\)](#) for more information about the tree structure)

The parameter \$requested_node is either an integer, indicating the user explicitly specified a node number in the page request, or null, indicating that the user did not explicitly specify a node. In the latter case the user may or may not have explicitly requested an area.

There are several cases we need to handle - if no node is explicitly requested, we need to identify the default page in the area - if the node is under embargo the node does not exist (from the POV of the user) - if the requested node is a section, we need to identify the default page in that section

void function main_index() [line 50]

main program for visitors

this routine is called from /index.php. It is the main program for visitors.

- **TODO** cleanup login/logout-code

bool|array function module_load_view(\$module_id) [line 479]

Function Parameters:

- *int* **\$module_id** indicates which module to load

load the visitor/view interface of a module in core

this includes the 'view'-part of a module via 'require_once()'. This routine first figures out if the view-script file actually exists before the file is included. Also, we look at a very specific location, namely: /program/modules/<modulename>/<module_view_script> where <modulename> is retrieved from the modules table in the database.

Note that if modulename would somehow be something like `"../..../etc/passwd\x00"`, we could be in trouble...

- **TODO** should we sanitise the modulename here? It is not user input, but it comes from the modules table in the database. However, if a module name would contain sequences of "../" we might be in trouble

bool function module_view(&\$theme, \$area_id, \$node_id, \$module_id) [line 436]

Function Parameters:

- *array* **&\$theme** a reference to the output object
- *int* **\$area_id** the area where we are looking for a node
- *int* **\$node_id** the node we are working with
- *int* **\$module_id** the module connected to the node we are working with

call the routine that generates the view (content) of module \$module_id

this loads the file containing the visitor interface for module \$module_id in core and subsequently calls the routine responsible for displaying the content (function modulename_view()). The routine module_view() is supposed to deposit any output into the \$theme via the appropriate methods such as \$theme->add_content().

void function update_statistics(\$node_id) [line 516]

Function Parameters:

- *int* **\$node_id** the page (node) that was viewed

update all statistics for the view of page \$node_id

this is a place for future extension. This routine is called once for every page view. It can be used to record relevant data in a table, for future reference, e.g.

- the IP-address of the visitor
- the \$node_id
- the current date/time
- the number of views of node \$node_id from the visitor's IP-address
- etc. etc.

Note that the table holding this information can quickly become very large. That requires some form of logrotate or condensing the data. This feature has yet to be developed.

- **TODO** maybe extend this routine to actually store more statistics information in a separate table

void function update_view_count(\$node_id) [line 529]

Function Parameters:

- *int \$node_id* the page (node) that need its view_count incremented with 1

update the view count for page \$node_id

manual.php

/program/manual.php - a kickstarter for the documentation

This script is an entry point; it can be called directly. It is also linked to from /program/admin.php, via the help button, implementing a context-sensitive help function. The following parameters are recognised:

- **language:** a language key, e.g. 'nl' (Dutch) or 'es' (Spanish). Default is 'en' (English)
- **topic:** one of the recognised topics, e.g. 'tools' or 'pagemanager'. Default is 'toc' (Table of contents)
- **subtopic:** one of the subtopics relevant in this topic, e.g. 'license' in the 'install' topic. Default is '' (None).

The actual work is done in the function [show_manual\(\)](#) below.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: manual.php,v 1.17 2016/06/28 14:08:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

array function get_available_languages(\$path) [line 278]

Function Parameters:

- **string \$path** is the directory where to look for languages (usually /program/languages).

construct a list of 0 or more languages from the languages directory

This routine examines the directory \$path to see which subdirectories exist. Each subdirectory indicates a possible language. An array keyed with these languages and the full name of the language in the language itself is returned (but it could be empty).

array function get_available_manuels(\$path) [line 248]

Function Parameters:

- **string \$path** is the directory where to look for manuals (usually /program/manuals).

construct a list of 0 or more languages of available manuals

This routine examines the directory \$path to see which subdirectories exist. Each subdirectory indicates a possible language. An array keyed with these languages and the full path to the directory containing the manual's index.html is returned (but it could be empty).

void function show_manual([\$language = 'en'], [\$topic = 'toc'], [\$subtopic = '']) [line 157]

Function Parameters:

- *string* **\$language** indicates the desired manual language
- *string* **\$topic** is the topic of interest to which we deep link
- *string* **\$subtopic** is a subtopic to allow for an even deeper link

redirect the user to a specific place in the manual OR show helpful message about downloading the manual

There is a Website@School Users' Guide available, in English. This is a separate download from the project's website. That means that it is optional to have the (English) manual installed. If it is installed, it is installed under /program/manuals/en/. There might also be translations available, say the Dutch version of the manual. That one would be installed in /program/manuals/nl/ which allows for peaceful co-existence of multiple translations of the manual. This script manual.php is designed to:

- redirect the user to the correct translation of the manual (if installed), and
- possibly use deep links to create context-sensitive help.

If the manual is not available in the requested language, the user is redirected to the English version (if that one IS installed). If no manual is installed at all, the user is shown a simple HTML-page which provides a link to the location where the manual(s) can be downloaded.

void function show_screen_choose_language(\$manuals) [line 357]

Function Parameters:

- *array* **\$manuals** holds a list of relative paths to manuals ToC's keyed by language code

show a screen to the visitor presenting a choice between various available translations of the manual

void function show_screen_download() *[line 307]*

**show a screen to the visitor hinting at downloading a manual archive from
download.websiteatschool.eu**

utf8lib.php

/program/lib/utf8lib.php - utility-routines for UTF-8

This file deals with the idiosyncrasies of UTF-8.

Reference: The Unicode Consortium, The Unicode Standard, Version 6.0.0, (Mountain View, CA: The Unicode Consortium, 2011, ISBN 978-1-936213-01-6) <<http://www.unicode.org/versions/Unicode6.0.0>>, chapter 3 (Conformance), page 94

Summary of the way valid code points are stored in 1 to 4 byte sequences.

bits	code-points	1st byte	2nd byte	3rd byte	4th byte
7	0000 0000 0000 0000	0xxx xxxx	0xxx.xxxx		
11	0000 0000 0000 0yyy yyxx xxxx	110y.yyyy	10xx.xxxx		
16	0000 0000 zzzz yyyy yyxx xxxx	1110.zzzz	10yy.yyyy	10xx.xxxx	
21	000w wwzz zzzz yyyy yyxx xxxx	1111.0www	10zz.zzzz	10yy.yyyy	10xx.xxxx

bits	byte 1	byte 2	byte 3	byte 4	comments
7	00 - 7F				U+0000 - U+007F
	80 - BF				--> ill-formed
	C0 - C1	80 - BF			--> overlong 2-byte
11	C2 - DF	80 - BF			U+0080 - U+07FF
	E0	80 - 9F*	80 - BF		--> overlong 3-byte
16	E0	A0 - BF*	80 - BF		U+0800 - U+0FFF
16	E1 - EC	80 - BF	80 - BF		U+1000 - U+CFFF
16	ED	80 - 9F*	80 - BF		U+D000 - U+D7FF
	ED	A0 - BF*	80 - BF		--> surrogates (U+D800 - U+DFFF)
16	EE - EF	80 - BF	80 - BF		U+E000 - U+FFFF
	F0	80 - 8F*	80 - BF	80 - BF	--> overlong 4-byte
21	F0	90 - BF*	80 - BF	80 - BF	U+10000 - U+3FFFF
21	F1 - F3	80 - BF	80 - BF	80 - BF	U+40000 - U+FFFFF
21	F4	80 - 8F*	80 - BF	80 - BF	U+100000 - U+10FFFF
	F4	90 - BF*	80 - BF	80 - BF	--> invalid planes 11 - 13
	F5 - F7	80 - BF	80 - BF	80 - BF	--> invalid planes 14 - 1F
	F8 - FB				--> invalid 5-byte sequence
	FC - FD				--> invalid 6-byte sequence
	FE - FF				--> disallowed (BOM)

Note: Non-standard continuation ranges are marked with * (only for byte 2)

- **Package** wascore

- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: utf8lib.php,v 1.8 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

zip.class.php

/program/lib/zip.class.php - create simple ZIP-archives

This file implements class Zip which allows for creating ZIP-archives on the fly

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: zip.class.php,v 1.6 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

version.php

'version.php' defines internal and external version numbers

The following constants are defined in this file:

- `WAS_VERSION` - the internal version number, e.g. 2008020100
- `WAS_RELEASE` - the external version number, e.g. 1.0 or 1.0.0
- `WAS_RELEASE_DATE` - the date that the distribution files were generated
- `WAS_ORIGINAL` - indicates the original (TRUE) or a modified version (FALSE) of this program

`WAS_VERSION` is used to see if the database version matches the program version. A difference between the two versions indicates an incomplete update. The version number is of the form `yyyymmddxx` where `yyyymmdd` is a date and the number `xx` is an auxiliary number that may or may not carry an extra meaning. `WAS_VERSION` is always greater than `WAS_VERSION` in a previous release of Website@School.

`WAS_RELEASE` is a free-format human-readable string indicating the the version of the program. It could take the form `major.minor` or `major.minor.patchlevel`.

`WAS_RELEASE_DATE` is the date on which the distribution package was generated. This date is set by editing this file `version.php` 'on the fly' from the `makedist.sh` script (see `/devel/tools/makedist.sh`).

`WAS_ORIGINAL` is a flag which indicates the original version (value TRUE) or a modified version (value FALSE) of the program. The License Agreement for Website@School states:

"In accordance with section 7(c) modified versions of the Program must clearly be marked in reasonable ways as different from the original version without misrepresenting the origin of the Program. This must be done by adding the phrase "Based on Website@School" to the Appropriate Legal Notices."

By defining `WAS_ORIGINAL` to FALSE, the phrase 'Powered by Website@School' in the interactive user interfaces will morph into 'Based on Website@School' automatically. The file `/program/about.html` should still be edited, though.

- **Package** wascore
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: version.php,v 1.28 2016/06/28 14:38:47 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

WAS_ORIGINAL = TRUE *[line 76]*

A boolean flag indicating this is either the original (TRUE) or a modified (FALSE) version of Website@School

WAS_RELEASE = 1.0.0 *[line 68]*

The external version number, like 1.0 or 1.0.0

WAS_RELEASE_DATE = 2016-06-29 *[line 72]*

Date of distribution file generation in ISO 8601 format: yyyy-mm-dd OR yyyy-mm-ddThh:mm:ss+0000

WAS_VERSION = 2016062900 *[line 64]*

The internal version number, like 2008012873 or 2008020100 (31 bits will work until the year 2147)

- **Used by** [error_exit\(\)](#) - indicate internal version in 'cryptic' message

Package wascore Classes

Class AclManager

[line 322]

class for manipulating (edit+save) access control lists

Overview

Every user account is associated with an access control list. This access control list boils down to a total of six tables in the database:

- acls
- acls_areas
- acls_nodes
- acls_modules
- acls_modules_areas
- acls_modules_nodes

These tables are defined as follows. acls:

acl_id serial*
permissions_jobs int
permissions_intranet int
permissions_modules int
permissions_nodes int

acls_areas:

acl_id int* (link to acls)
area_id int* (link to areas)
permissions_intranet int
permissions_modules int
permissions_nodes int

acls_nodes:

acl_id int* (link to acls)
node_id int* (link to nodes)
permissions_modules int
permissions_nodes int

acls_modules:
acl_id int* (link to acls)
module_id int* (link to modules)
permissions_modules int

acls_modules_areas:
acl_id int* (link to acls)
module_id int* (link to modules)
area_id int* (link to areas)
permissions_modules int

acls_modules_nodes:
acl_id int* (link to acls)
module_id int* (link to modules)
node_id int* (link to nodes)
permissions_modules int

*marked fields are (part of) the primary key

The six tables mentioned above deal with the following permission bitmasks.

- permissions_jobs
- permissions_intranet
- permissions_modules
- permissions_nodes

The reasons to split these permission masks into six tables are:

1. Some permissions only apply to the site-level and it makes no sense to specify them for a particular combination of area, node or module. Example: permissions_jobs.
2. Some permissions can be granted for current and future objects. Example: permissions_intranet. If these permissions are granted at the site level (in table acls), then they apply not only to all current protected areas but also to all future protected areas. The same permissions could be granted on a per-area-basis but that might require adjusting the permissions once a new protected area is added to the site.
3. Sometimes it is more convenient to specify the permissions on a higher level because otherwise the size of the database may get out of hand. Example: if every user has a permission bitmask for every node on the site, the corresponding acl would have number_of_users x number_of_nodes entries. That is completely unmanageable, even for small to medium size sites.

Users and group/capacities

A user can also participate in a group in a particular capacity, e.g. member of group 'grade8' in the 'pupil'- or the 'teacher'-capacity. Every combination of group and capacity (eg

'grade8/pupil') is also associated with an access control list.

The full access control list for a user is the combination of the ACL directly associated with the user account and the ACLs associated with the group/capacities that apply to the user account. The effective permissions for a user are the result of OR'ing the permissions of all ACLs.

A specific permission is always indicated by a bit set to '1'. If a particular bit is set to '0', the user does not have the corresponding permission. This implies that the (special) bitmask 0 (zero, 32 bits are all not set) corresponds to 'no permissions at all'. It also implies that the (special) bitmask -1 (minus one, 32 bits are all set) equates to 'all permissions'.

Therefore, the **easy** way to grant access is to set the permissions bitmask to -1. This is the so-called Guru-option or -role or the Guru-permissions. However, note that granting a user or a group/capacity Guru-permissions, means that that user (these users) can do serious harm to the system because she (they) are allowed to do anything. The **safe** way is to grant as few permissions as possible.

Roles

In order to make it easier to setup the access controls and stay away from directly manipulating individual bits in a bitmask the various permission bits are combined into roles.

Two roles are always available for selection:

- permissions == 0: ROLE_NONE
- permissions == -1: ROLE_GURU

Defining other roles is done at the appropriate place, e.g. inside the code for a module.

Example: suppose that there is a module called 'Forum' which works with authenticated users. Depending on this module's permission bits the users are allowed to perform certain actions, e.g.

- read messages in the forum (bit 0, value 1)
- write messages in the forum (bit 1, value 2)
- edit their own messages (bit 2, value 4)
- edit other users' messages (bit 3, value 8)
- manage useraccounts for the forum (bit 4, value 8)

This leads to many possible combinations of set and reset bits. However, it is more practical to combine these bits into a few roles with a descriptive name:

- permissions = 1: ROLE_FORUM_VISITOR => "Visitor"
- permissions = 1+2+4 = 7: ROLE_FORUM_MEMBER => "Member"
- permissions = 1+2+4+8 = 15: ROLE_FORUM_MODERATOR => "Moderator"
- permissions = 1+2+4+8+16 = 31: ROLE_FORUM_ADMINISTRATOR => "Administrator"

By using these symbolic names for certain combinations of bitmasks it becomes easier to

manage many users and many forums without having to know what every bit means, exactly.

Obviously these roles (defined via the module in this example) will end up in a dropdown list where the appropriate role can be assigned.

Note that it is not necessary to have hierarchical roles as demonstrated in this example. It is very well possible to define two roles that must work together: `ROLE_EDITOR` could be a bitmask that allows for adding (1), editing (2), deleting (4), previewing (8) news articles whereas `ROLE_PUBLISHER` could be limited to previewing (8) and publishing (16) news articles, but not editing them. That would make sure that at least two different people are required to create and publish an article. (However, any 'Guru', with all permissions granted due to the -1 bitmask, could create + publish articles by herself.)

Module-permissions in `acls`, `acls_areas` and `acls_nodes`

The fields `permissions_modules` in the tables `acls`, `acls_areas` and `acls_nodes` should be considered as 'blanket permissions'. If a permission is set in either of these tables, the permissions apply to **all** modules at site level (`acls`), area_level (`acls_areas`) or node_level (`acls_nodes`).

Because these permissions apply to **all** modules, the only realistic roles in these cases can be either `ROLE_NONE` (permissions = 0) or `ROLE_GURU` (permissions = -1). Any other role could be meaningless for one or more modules.

Furthermore, it is a little over the top to specify permissions for **all** modules in a particular node. (It almost doesn't make sense). Therefore, the corresponding dialog only deals with these two roles `ROLE_NONE` and `ROLE_GURU` at the site level and the area level. The node level is not used for modules (but it is for pagemanager permissions - the field `permissions_nodes` - at the node level).

Typical usage

Example 1: displaying a dialog with intranet permissions for a group

```
$acl = new AclManager($output,$acl_id,ACL_TYPE_INTRANET);
$acl->set_action(array('job'=>'accountmanager','task'=>'groupsave','group'=>'8'));
$acl->set_dialog(GROUPMANAGER_DIALOG_INTRANET);
$acl->show_dialog();
```

... The result of this snippet is that a complete dialog is output to the content area of the `$output` object, including the current values from the database. The whole dialog is wrapped in a `FORM`-tag with action property based in the array set with the `set_action()` method. The dialog is POSTed with either a Save, a Done or a Cancel button.

Example 2: saving the data for the intranet permissions for a group

```
$acl = new AclManager($output,$acl_id,ACL_TYPE_INTRANET);
$acl->set_action(array('job'=>'accountmanager','task'=>'groupsave','group'=>'8'));
```

```

$acl->set_dialog($dialog);
if (!$acl->save_data()) {
    $acl->show_dialog(); // redo dialog, but without a distracting menu this time
    return;
}
...

```

The effect of this snippet is that an attempt is done to validate and save the data as it was POSTed (ie: the new values are available in `$_POST[]`). If, however, saving the data did not work, the dialog is displayed again, this time using the data from `$_POST[]` rather than from the database.

Example 3: displaying a dialog with admin permissions for a user

```

$related_acls = array($acl_id1 => "group1/capacity1", $acl_id2 => "group2/capacity2", ...);
$acl = new AclManager($this->output, $acl_id, ACL_TYPE_ADMIN);
$acl->set_related_acls($related_acls);
$acl->set_action(array('job'=>'accountmanager', 'task'=>'usersave', 'user'=>'23'));
$acl->set_dialog(USERMANAGER_DIALOG_ADMIN);
$acl->show_dialog();

```

This is comparable to example 1. The difference is that in a User-ACL there is an option to display existing permissions from the user's group/capacities. This information is displayed in the third column in the dialog. This provides a clue for the user that certain permissions might already be granted to the user via a group membership. The related permissions are communicated via an array with (integer) `acl_id`'s as key and a string value identifying the group/capacity.

- **Package** wascore
- **TODO** there is something not right with buffering the tabledefs. If an error occurs, we get FALSE instead of an array. Mmmmm....

AclManager::\$acl_id

int = 0 [line 330]

- **Var** `$acl_id` identifies the ACL we are dealing with

AcIManager::\$acl_type

int = 0 [*line 333*]

- **Var** \$acl_type identifies the type of ACL we are dealing with

AcIManager::\$area_view_areas_open

array|bool = FALSE [*line 374*]

- **Var** \$area_view_areas_open identifies which areas are currently 'open' and 'closed'

AcIManager::\$area_view_a_params

array = NULL [*line 371*]

- **Var** \$area_view_a_params holds the parameters for linking to opening/closing an area

AcIManager::\$area_view_enabled

bool = FALSE [*line 377*]

- **Var** \$area_view_enabled if TRUE we add icons to areas so they can expand/collapse (default=FALSE)

AcIManager::\$a_params_save

array|null = NULL [*line 339*]

- **Var** \$a_params_save holds the parameters for the action property of the HTML-form that is created

AcIManager::\$dialog

int = 0 [line 348]

- **Var** \$dialog identifies the exact dialog and it is added to the dialog as hidden field

AcIManager::\$dialogdef

array = NULL [line 388]

- **Var** \$dialogdef holds the current dialogdef, maybe including error messages from a failed validation

AcIManager::\$dialogdef_areas

array = array() [line 395]

- **Var** \$dialogdef_areas holds information of zero or more areas and the number of contained nodes

AcIManager::\$dialogdef_areas_total

int = NULL [line 392]

- **Var** \$dialogdef_areas_total holds the total number of items that could be displayed in the dialogdef

AcIManager::\$header

string = [line 342]

- **Var** \$header the title of the dialog, displayed at the top of the content area

AcIManager::\$intro

string = [line 345]

- **Var** \$intro the introductory text for the dialog, displayed below the \$header

AcIManager::\$output

object|null = NULL [line 327]

- **Var** collects the html output

AcIManager::\$pagination_a_params

array = NULL [line 355]

- **Var** \$pagination_a_params holds the parameters for linking to another view of the dialog

AcIManager::\$pagination_enabled

bool = FALSE [line 364]

- **Var** \$pagination_enabled if TRUE we do try to paginate the display (default=FALSE)

AcIManager::\$pagination_limit

int = NULL [line 358]

- **Var** \$pagination_limit the preferred size of a screenfull of dialog lines

AcIManager::\$pagination_offset

int = NULL [line 361]

- **Var** \$pagination_offset the record where the current screen begins

AcIManager::\$pagination_total

bool = 0 [line 384]

- **Var** \$pagination_total holds the total number of elements to display

AcIManager::\$related_acls

array = NULL [line 336]

- **Var** \$related_acls if not NULL identifies a list of acl_id => 'description' pairs with related ACLs

Constructor *void* function AcIManager::AcIManager(&\$output, \$acl_id, \$acl_type) [line 408]

Function Parameters:

- *object* **&\$output** holds the output that eventually is send to the user's browser
- *int* **\$acl_id** identifies the ACL we are dealing with (primary key in acls table)
- *int* **\$acl_type** identifies the type of ACL we are dealing with, e.g. ACL_TYPE_INTRANET or ACL_TYPE_ADMIN

constructor for the AcIManager

this constructs a new AcIManager object. Essential information such as the acl_id and the

acl_type are stored, for future reference.

int/bool function AclManager::calc_areas_total(&\$areas) [*line 902*]

Function Parameters:

- **array &\$areas** an array with summary information about areas, including the # of nodes to show

calculate the total number of items (site, areas, nodes) to show in dialog

the 'open' or 'closed' status of an area is dictated by \$open_areas:

- if \$open_areas is an array the elements look like \$area_id => \$show, where
\$show == TRUE indicates the area is 'open' and \$show == FALSE indicates the area is 'closed'
- if \$open_area is a boolean and the value is TRUE. _all_ areas are to be considered 'open'
- otherwise _all_ areas are to be considered 'closed'.

The returned value \$total is the sum of the number of areas and the number of 'showable' nodes (as per the information in \$open_areas). If there are no areas at all, \$total is 0. If an error occurs, this routine returns FALSE.

The parameter \$areas is used as a return value. It is keyed with \$area_id and filled with pertinent information about the areas:

- int \$area_id: the number of the area (also the key of the the \$areas array)
 - string \$title the name of the area
 - bool \$is_active indicating an active area (TRUE) or an inactive area (FALSE)
 - int \$nodes the total number of nodes in this area (could be 0 if no nodes were added yet)
 - int permissions_nodes the bitmap containing the existing node permissions for this area
- Also, the following information is added to the resulting array:
- bool \$show if TRUE, all nodes in this area should be displayed
 - int \$first indicating the offset of the row for this area, relative from the start of the list of areas
 - int \$last indicating the offset of the last item to show in this area (could be the same as \$first)

The latter three values are used to skip \$offset rows when constructing the dialog.

- **Uses \$DB;**

array function AclManager::dialog_tableform(\$href, &\$dialogdef, [\$show_related = FALSE]) [*line 1465*]

Function Parameters:

- *string* **\$href** the target of the HTML form
- *array* **&\$dialogdef** the array which describes the complete dialog
- **\$show_related**

construct a form with a dialog in a table with 2 or 3 columns

this constructs a 2- or 3-column table and fills it with data from the dialogdef.

The first column holds the labels for the widgets. The second column holds the corresponding widget, e.g. a list box with roles. The optional third column (depends on the flag **\$show_related**) shows related information. This is used to list group/capacities and roles from related groups (ie. groups of which the user is a member).

The table has headers for the columns: 'Realm','Role' and optional 'Related'. Rows in the table can have alternating colours via the odd/even class. This is done via the stylesheet.

- **TODO** bailing out on non-array is a crude way of error handling: this needs to be fixed

void function **AcIManager::enable_area_view**(\$a_params, \$areas_open) [*line 532*]

Function Parameters:

- *array* **\$a_params** basic parameters (excluding \$area) that lead to the page where expand/collapse is processed
- *array|bool* **\$areas_open** indicator(s) for 'open' and 'closed' areas

further initialise the AcIManager and enable the area expand/collapse feature

this stores the necessary information about 'open' and 'closed' areas. The parameter **\$areas_open** indicates the current state of affairs: (**\$areas_open === FALSE**) means all areas are closed (**\$areas_open === TRUE**) means all areas are opened If **\$areas_open** is an array, it contains **area_id**'s as key and **TRUE** or **FALSE** as value. A value of **TRUE** indicates that an area is currently 'open', **FALS** or no value set means 'closed'.

The parameters in **\$a_params** combined with **\$WAS_SCRIPT_NAME** yield an URL where the changes are processed.

void function AclManager::enable_pagination(\$a_params, \$limit, \$offset) [line 504]

Function Parameters:

- *array* **\$a_params** basic parameters (excluding \$offset and \$limit) that lead to the correct page
- *int* **\$limit** the preferred size of a screenfull of dialog lines
- *int* **\$offset** the record where the current screen begins

further initialise the AclManager and enable the dialog pagination feature

this stores the information that is necessary when a dialog has to be broken up into two or more screens (via the pagination facility in \$output). This routine stores the essential information such as the parameters that lead to the correct page (in \$a_params) and the current offset. The other necessary parameters are calculated dynamically before add_pagination() is called.

Note that pagination is only enabled after this routine is called at least once; by default we do NOT do pagination. (Actually: pagination is only used in the acl_types ACL_TYPE_PAGEMANAGER and ACL_TYPE_MODULE).

- **Uses** \$CFG;

array function AclManager::get_dialogdef_admin(\$acl_id, [\$related_acls = NULL]) [line 1145]

Function Parameters:

- *int* **\$acl_id** the acl of interest
- *array* **\$related_acls** NULL or a list of acl_id => "group/capacity" pairs for related permissions

construct an array with the admin dialog information

this creates an array with widgets for all possible admin jobs for \$acl_id.

This dialog is supposed to be rendered as a 2-column (group acl) or 3 column (user acl) table. The contents of the 3rd column is a list (an array) of related permissions, ie. the permissions a user has been granted via a group membership. The related information is stored in an extra array element 'related'.

The related information is constructed only in the case where \$related_acls is not NULL.

The dialog is filled with the current values via `$item['value']` but as a side effect the current value is also recorded in `$item['old_value']`). This makes it easier to determine whether any values have changed (see [save_data_admin\(\)](#)).

- **TODO** handle the related information in this dialog

array function `AclManager::get_dialogdef_intranet($acl_id, [$related_acls = NULL])` [line 1026]

Function Parameters:

- *int* **\$acl_id** the acl of interest
- *array* **\$related_acls** NULL or a list of `acl_id => "group/capacity"` pairs for related permissions

construct an array with the intranet dialog information

this creates an array with 1 or more list boxes with the current roles for `$acl_id` for intranet access at the site level and for individual private areas. This dialog is supposed to be rendered as a 2-column (group acl) or 3 column (user acl) table. The contents of the 3rd column is a list (an array) of related permissions, ie. the permissions a user has been granted via a group membership. The related information is stored in an extra array element 'related'.

The related information is constructed only in the case where `$related_acls` is not NULL.

The dialog is filled with the current values via `$item['value']` but as a side effect the current value is also recorded in `$item['old_value']`). This makes it easier to determine whether any values have changed (see `save_data_internet()`).

- **TODO** handle the related information in this dialog

void function `AclManager::get_dialogdef_pagemanager($acl_id, $related_acls)` [line 1240]

Function Parameters:

- **\$acl_id**
- **\$related_acls**

construct a dialog definition for pagemanager permissions

string function AclManager::get_icon_area(\$area_id, \$area_is_open, \$offset) [*line 1886*]

Function Parameters:

- *int* **\$area_id** the area to open/close (0 means: open site level)
- *bool* **\$area_is_open** current status
- *int* **\$offset** the position of this icon in the current list of items

construct a clickable icon to open/close this area

This is a toggle: if the area is closed the closed icon is shown, but the action in the A-tag is to open the icon (and vice versa).

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER
- **Uses** \$CFG

array function AclManager::get_permissions(\$acl_id, [\$related_acls = NULL]) [*line 1559*]

Function Parameters:

- *int* **\$acl_id** the primary acl_id (used for both users and groups)
- *array|null* **\$related_acls** an array with related acls for this user or NULL for group acls

retrieve an array with 0, 1 or more records with permissions from table 'acls'

this constructs an array with all (or selected) permissions from the 'acls' table for the specified acl \$acl_id and optionally for all related acl_id's in \$related_acls. The resulting array is keyed by acl_id.

array function AclManager::get_permissions_areas(\$acl_id, [\$related_acls = NULL], [\$areas = NULL]) [*line 1593*]
Function Parameters:

- *int* **\$acl_id** the primary acl_id (used for both users and groups)
- *array|null* **\$related_acls** an array with related acls for this user keyed by 'acl_id' or NULL for group acls
- *array|null* **\$areas** an array with areas of interest keyed by 'area_id' or NULL for all areas

retrieve an array with 0, 1 or more records with permissions from table 'acls_areas'

this constructs an array with all permissions from the 'acls_areas' table for the specified acl \$acl_id and optionally for all related acl_id's in \$related_acls and optional areas. The resulting array is keyed by area_id and acl_id.

Note that by making the result keyed by area_id first (and then acl_id) it becomes possible to step through a list of areas and have 0,1 or more acls for that area in a single array, e.g. \$acls = \$permissions[16] yields the selected acls that apply to area 16. That is handy when constructing dialogs iterating through areas such as intranet permissions.

array function AclManager::get_permissions_nodes_in_area(\$area_id, \$acl_id, [\$related_acls = NULL]) [*line 1648*]
Function Parameters:

- *array* **\$area_id** the area where the nodes reside
- *int* **\$acl_id** the primary acl_id (used for both users and groups)
- *array|null* **\$related_acls** an array with related acls for this user keyed by 'acl_id' or NULL for group acls

retrieve an array with 0, 1 or more records with permissions from table 'acls_nodes'

this constructs an array with all permissions from the 'acls_nodes' table for the specified acl \$acl_id and optionally for all related acl_id's in \$related_acls and optional nodes. The resulting array is keyed by node_id and acl_id.

Note that by making the result keyed by node_id first (and then acl_id) it becomes possible to step through a list of nodes and have 0,1 or more acls for that node in a single array, e.g. \$acls = \$permissions[16] yields the selected acls that apply to node 16. That is handy when constructing dialogs iterating through nodes such as pagemanager permissions.

array function AclManager::get_roles_intranet() [line 1384]

construct an option list with roles for intranet access

array function AclManager::get_roles_pagemanager([\$level = ACL_LEVEL_NONE]) [line 1402]

Function Parameters:

- *int \$level* limits permissions to level 'page', 'section', 'area' or 'site'

construct an option list with roles for pagemanager access

bool function AclManager::save_data() [line 579]

save the changed data for the selected acl_type

this interprets the data from the selected dialog and saves the (changed) permission data accordingly. This, too, is merely a dispatcher to the subroutines that do the actual work.

bool function AclManager::save_data_admin() [line 765]

save changed job permissions to the database

this saves the changed job permissions to the acls table.

If the user selected the guru option, we simply set the permissions to JOB_PERMISSION_GURU (i.e. all permissions set). If not, we iterate through all existing permissions and set the corresponding bits. Then the data is saved to the correct acls-record. After that we re-read the dialogdef again because the user may want to continue editing rather than be [Done] with job permissions.

- **TODO** fix the crude error check on dialogdef === FALSE here

bool function AclManager::save_data_intranet() [line 640]

save the changed roles for intranet access to the tables 'acls' and 'acls_areas'

this interprets the data from the intranet dialog and saves the changed roles accordingly

bool function AclManager::save_data_pagemanager() [line 989]

save the changed roles for pagemanager to the tables 'acls' and 'acls_areas' and 'acls_nodes'

this interprets the data from the pagemanager dialog and saves the changed roles

accordingly

bool function AclManager::save_data_permissions() [line 655]

save the changed roles in the dialog to the corresponding tables 'acls'

this interprets the data from the current dialog and saves the changed roles accordingly. Note that the information about tables and fields etc. is all contained in the dialogdef so we can use this generic save_data() routine.

void function AclManager::set_action([\$a_params = NULL]) [line 439]

Function Parameters:

- *array \$a_params*

further initialise the AclManager with the dialog action property

this stores an array with parameters that must be added to the action property of the HTML form that will be POSTed, i.e. the URL to which the dialog will be posted. Example of such an array is: array('job' => 'accountmanager', 'task' => 'user_save', 'user' => 123); The information in this array is later combined with WAS_SCRIPT_NAME.

void function AclManager::set_dialog([\$dialog = 0]) [line 480]

Function Parameters:

- *int \$dialog* a unique identification (within this job) of the dialog

further initialise the AclManager with the dialog identification

this stores an integer number that is used to identify the dialog. This number is subsequently added to the dialog as a hidden field, which makes it possible to identify the dialog once it is POSTed

void function AclManager::set_header([\$header = ""]) [line 453]

Function Parameters:

- *string \$header* text to show as title

further initialise the AclManager with the dialog header

this stores a string that is used as a title for the dialog Note that this header may be extended with a (translated) string like '{FIRST}-{LAST} of {TOTAL}' in case of a paginated display.

void function AclManager::set_intro([\$intro = "]) [line 466]

Function Parameters:

- *string \$intro* introductory text for the dialog

further initialise the AclManager with the dialog introductory text

this stores a string that is displayed after the dialog header. This text supposedly contains some more information about the dialog.

void function AclManager::set_related_acls([\$related_acls = NULL]) [line 423]

Function Parameters:

- *array \$related_acls* identifies a list with related ACLs

further initialise the AclManager with related Acl's

this stores the array with 0, 1 or more key-value-pairs of the form \$acl_id => \$group_capacity_name, e.g. 3 => 'staff/member', 4 => 'grade7/teacher'

void function AclManager::show_dialog() [line 547]

show the dialog where the selected Acl can be modified

this shows the dialog corresponding to the acl_type that was previously selected, including existing data from the previously selected acl_id. Note that this routine is only a simple dispatcher; actual work is done in subroutines.

void function AclManager::show_dialog_admin() [line 738]

display a tabular form for manipulating admin permissions

This dialog is a table consisting of 2 (group acl) or 3 (user acl) columns. The first column holds the various job names/descriptions. The second column holds a checkbox for the job. The optional third column holds corresponding (existing) permissions based on a group/capacity membership of the user. This 3rd column is displayed only when there are related acls (indicated via related_acls not empty)

void function `AcIManager::show_dialog_intranet()` [*line 620*]

display a tabular form for manipulating intranet permissions

This dialog is a table consisting of 2 (group acl) or 3 (user acl) columns. The first column holds the text 'All areas' or the name of a private area (if any) The second column holds a listbox where the user can select 1 out of 3 roles: 0 = "--", 1 = "Access", -1 = "Guru". The optional third column holds corresponding (existing) roles based on a group/capacity membership of the user. This 3rd column is displayed only when there are related acls (indicated via `related_acls` not empty)

void function `AcIManager::show_dialog_pagemanager()` [*line 842*]

display a tabular form for manipulating pagemanager permissions

This dialog is a table consisting of 2 (group acl) or 3 (user acl) columns. The first column identifies the site, areas or nodes within areas. The second column holds a listbox where the user can select a role for that particular item. The roles 0 = "--" and -1 = "Guru" are always available. The optional third column holds corresponding (existing) roles based on a group/capacity membership of the user. This 3rd column is displayed only when there are related acls (indicated by `$related_acls` not being empty).

The main purpose of this routine is to show some `$this->pagination_limit` table rows (starting at `$this->pagination_offset`) and corresponding [Save], [Done] and [Cancel] buttons that eventually lead to the save routine. (If pagination is not enabled, the full overview is displayed).

Note that the actual pagination is performed in [get_dialogdef_pagemanager\(\)](#). The additional feature of expanding/collapsing areas in the display is also done in [get_dialogdef_pagemanager\(\)](#).

- **Uses** `$WAS_SCRIPT_NAME`
- **Uses** `$CFG`;

void function `AcIManager::show_tree_walk(&$dialogdef, &$tree, &$permissions_nodes, &$index, $node_id, $first, $last, $acl_id, &$related_acls)` [*line 1799*]

Function Parameters:

- *array* **&\$dialogdef** collects the widgets
- *array* **&\$tree** a reference to the complete tree built earlier
- *array* **&\$permissions_nodes** contains permissions per node

- *int* **&\$index** only add node to dialogdef if \$index is between \$first and \$last, increments for every node
- *int* **\$node_id** the first node of this tree level to show
- *int* **\$first** lower bound of interval
- *int* **\$last** upper bound of interval
- *int* **\$acl_id** the acl we are rendering
- *array|null* **&\$related_acls** an array with related acls for this user keyed by 'acl_id' or NULL for group acls

add the specified node to dialogdef, optionally all subtrees, and subsequently all siblings

this routine adds a widget to the dialogdef for the specified node After that, any subtrees of this node are added too, using recursion This continues for all siblings of the specified node until there are no more (indicated by a sibling_id equal to zero).

- **Usedby** [AclManager::show_tree_walk\(\)](#)
- **Uses** [AclManager::show_tree_walk\(\)](#)
- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER
- **Uses** \$CFG

array function AclManager::tree_build(\$area_id, \$acl_id, \$related_acls) [*line 1707*]

Function Parameters:

- *int* **\$area_id** the area for which to build the tree
- *int* **\$acl_id** the primary acl_id (used for both users and groups)
- *array|null* **\$related_acls** an array with related acls for this user keyed by 'acl_id' or NULL for group acls

build a tree of all nodes in an area

this routine constructs a tree-structure of all nodes in area \$area_id in much the same way as [tree_build\(\)](#) does. However, in this routine we keep the cargo limited to a minimum: the fields we retrieve from the nodes table and store in the tree are:

- node_id
- parent_id
- is_page
- title
- link_text
- module_id

Also, the tree is not cached because that does not make sense here: we only use it to construct a dialogdef and that is a one-time operation too.

Class AdminOutput

[line 618]

conveniently collect output

This class allows for a convenient way to temporarily store output, in random order and still being able to output to the browser in the correct order (eg. headers() first, etc.).

This class 'knows' everything about the structure of a generated page. Most of this knowledge is contained in \$this->get_html(); The actual layout is defined in the corresponding stylesheets, e.g. admin_base.css.

Typical use of this object is to add HTML-code to various parts of the page via the add_*() methods and finally sending the collected output to the user's browser with \$this->send_output(). That's it.

- **Package** wascore
- **TODO** carefully check if we need more headers in html-head section of document, see [AdminOutput\(\)](#).
- **TODO** add a 'funnel mode': disable all distracting links that could seduce the user to leave and leave locked records (eg. nodes)

AdminOutput::\$breadcrumbs

array = array() [line 659]

- **Var** a list URL components etc. identifying the path that leads to the current screen

AdminOutput::\$content

array = array() [line 638]

- **Var** collection of items/lines that are part of the content area

AdminOutput::\$dtd

*string = <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">* [line 620]

- **Var** the standard doctype (default: HTML 4.01 Transitional)

AdminOutput::\$funnel_mode

bool = FALSE [line 665]

- **Var** if TRUE, the display should be in funnel-mode, ie. distracting links should be non-working

AdminOutput::\$helptopic

string = [line 656]

- **Var** the additional parameter to add to the help link in the navigation

AdminOutput::\$html_head

array = array() [line 632]

- **Var** collection of items/lines that will be output as part of the HTML-head section

AdminOutput::\$http_headers

array = array() [line 629]

- **Var** collection of individual http-headers that are to be sent *_before_* any HTML is sent

AdminOutput::\$menu

array = array() [line 635]

- **Var** collection of items/lines that are part of the menu (could be empty)

AdminOutput::\$messages_bottom

array = array() [line 647]

- **Var** collection of messages that are to be displayed via a javascript alert() at END of page

AdminOutput::\$messages_inline

array = array() [line 641]

- **Var** collection of messages that are to be displayed just below the navigation bar

AdminOutput::\$messages_top

array = array() [line 644]

- **Var** collection of messages that are to be displayed via a javascript alert() at START of page

AdminOutput::\$pagination

array = array() [line 662]

- **Var** a list URL components etc. comprising a list of links to paginated display of lists

AdminOutput::\$skin

AdminSkin = [line 653]

- **Var** the currently selected skin

AdminOutput::\$subtitle

string = [line 626]

- **Var** a subtitle displayed in the generated page (for the time being it is empty or has a)

AdminOutput::\$suppress_output

bool = FALSE [line 668]

- **Var** if TRUE, no output should be sent whatsoever

AdminOutput::\$text_only

bool = FALSE [line 650]

- **Var** this switches the navigation between image-based and text-based

AdminOutput::\$title

string = [line 623]

- **Var** the title to display in both the title tag and in the page itself (usually the sitename)

Constructor *void* function AdminOutput::AdminOutput([\$skin = 'base'], [\$title = ''], [\$subtitle = '']) *[line 682]*

Function Parameters:

- *string* **\$title** the title to display in both the title tag and inside the document
- *string* **\$subtitle** the text to display underneath the title in the document (or if empty)
- **\$skin**

constructor

This sets up the object and adds the title and subtitle.

- **TODO** is it really wise to add a base header? It interferes with the session cookie whenever you login at another URL than the base+'admin.php'... Comment it out for now
- **TODO** do we need a link rel="shortcut icon" type of header too?
- **TODO** do we really need more meta-headers?

void function AdminOutput::add_breadcrumb(\$href, [\$params = NULL], [\$attributes = NULL], [\$anchor = NULL])
[line 1565]

Function Parameters:

- *string* **\$href** holds the hypertext reference
- *string|array* **\$params** holds the parameters to add to the \$href
- *string|array* **\$attributes** holds the attributes to add to the tag
- *string* **\$anchor** the text that identifies the breadcrumb

add a breadcrumb to the breadcrumb trail

this stores information about a crumb in the breadcrumb trail into the breadcrumbs array. We store this information in pieces that are readily usable with [html_a\(\)](#). However, by constructing the links at the latest possible time, we are able to suppress the real links, by replacing the href + parameters with a bare "#". This trick can be used to keep the user focussed on the task at hand (in 'funnel-mode').

void function AdminOutput::add_content(\$content) [line 1522]

Function Parameters:

- *string|array* **\$content** the line(s) of text to add

add a line or array of lines to the content part of the document

void function AdminOutput::add_html_header(\$headerline) [line 1436]

Function Parameters:

- *string* **\$headerline** headerline to add

add a header to the HTML head part of the document

void function AdminOutput::add_http_header(\$headerline) [line 1427]

Function Parameters:

- *string* **\$headerline** headerline to add

add an HTTP-header

void function AdminOutput::add_menu(\$menuline) [line 1535]

Function Parameters:

- *string* **\$menuline** the line of text to add

add a line to the menu part of the document

void function AdminOutput::add_message(\$message) [line 1474]

Function Parameters:

- *string/array* **\$message** message(s) to add inline

add a message to the list of inline messages, part of the BODY of the document

void function AdminOutput::add_meta(\$meta) [line 1499]

Function Parameters:

- *array* **\$meta** an array with name-value-pairs that should be added to the HTML head part

add a line with meta-information to the HTML head part of the document

void function AdminOutput::add_meta_http_equiv(\$meta) [line 1511]

Function Parameters:

- *array* **\$meta** an array with name-value-pairs that should be added to the HTML head part

add a line with http-equiv meta-information to the HTML head part of the document

int function AdminOutput::add_pagination(\$href, \$base_params, \$num_records, \$limit, \$current_offset, \$num_links) [line 1623]

Function Parameters:

- *string* **\$href** the script to call
- *array* **\$base_params** the necessary parameters to land on the correct page
- *int* **\$num_records** the total length of the list
- *int* **\$limit** the preferred size of the screen to show
- *int* **\$current_offset** the record that starts the current screen
- *int* **\$num_links** the maximum number of pages to show in the navbar (excluding Prev/Next/All)

add a pagination navigation bar to the output

this adds a pagination navbar to the output making it easier to step through a long listing of items a screen at a time

Features:

- the 'Prev' and 'Next' buttons do wrap: whenever we hit the begin/end of the list, we start again at the end/begin.
- if there are more screens to show than \$num_links, we show at most \$num_links links whenever we are NOT at the start of the list, the smallest link is displayed differently (via translation), e.g. via a left bracket '<' or three dots '...'
- the same trick is used at the end of the list to indicate there's more (via a different translation), e.g. via a right bracket '>' or three dots '...'

This function returns the number of screens that is required to show all the items in screens of at most \$limit items. The number of screens is at least 1.

void function AdminOutput::add_pagination_item(\$href, [\$params = NULL], [\$attributes = NULL], [\$anchor = NULL]) [*line 1589*]

Function Parameters:

- *string* **\$href** holds the hypertext reference
- *string/array* **\$params** holds the parameters to add to the \$href
- *string/array* **\$attributes** holds the attributes to add to the tag
- *string* **\$anchor** the text that identifies the link

add a link to screen of a paginated list to the existing list

this stores information about a screen in the paginated display of a list. We store this information in pieces that are readily usable with [html_a\(\)](#). However, by constructing the links at the latest possible time, we are able to suppress the real links, by replacing the href + parameters with a bare "#". This trick can be used to keep the user focussed on the task at hand (in 'funnel-mode').

void function AdminOutput::add_popup_bottom(\$message) [line 1460]

Function Parameters:

- *string|array* **\$message** message(s) to add

add a message to the list of popup-messages at the BOTTOM of the document

void function AdminOutput::add_popup_top(\$message) [line 1446]

Function Parameters:

- *string|array* **\$message** message(s) to add

add a message to the list of popup-messages at the TOP of the document

void function AdminOutput::add_stylesheet(\$url) [line 1488]

Function Parameters:

- *string* **\$url** url of the stylesheet

add a link to a stylesheet to the HTML head part of the document

string function AdminOutput::get_address([\$m = ""]) [line 1356]

Function Parameters:

- *string* **\$m** left margin for increased readability

return the reconstructed URL in a single (indented) line

actual code moved to waslib; keep this wrapper for compatibility

string function AdminOutput::get_bottomline([\$m = ""]) [line 1335]

Function Parameters:

- *string \$m* left margin for increased readability

report basic performance indicators in a single line

This calculates the execution time of the script and the number of queries. Note a special trick: we retrieve the translated string in a dummy variable before calculating the number of queries because otherwise we might miss one or more query from the language/translation subsystem.

Note that the message containing the performance indicators is only generated when debug is TRUE; the information is not that interesting for ordinary users.

string function AdminOutput::get_breadcrumbs([\$m = ""]) [line 1371]

Function Parameters:

- *string \$m* left margin for increased readability

retrieve/construct a list of 0 or more clickable breadcrumbs

this reads the breadcrumbs-array and constructs the breadcrumb trail. If \$this->funnel_mode is TRUE, the links are still clickable but the href-part is replaced with a "#" instead of the 'real' href + parameters.

string function AdminOutput::get_content([\$m = ""]) [line 926]

Function Parameters:

- *string \$m* left margin for increased readability

get all lines in the content DIV in a single properly indented string

string function AdminOutput::get_div_messages(\$bullets, [\$m = ""]) [line 983]

Function Parameters:

- *bool \$bullets* if TRUE then _all_ messages get a bullet
- *string \$m* left margin for increased readability

get a perhaps bulleted list of messages in a DIV

This constructs an unordered list with messages, if there are any. If there is no message at all, an empty string is returned (without DIV). Previously, if there was a single message, no bullet was added to the message. If there were two or more messages, bullets were added. We now (april 2012) have a boolean parameter `$bullets` which modifies this behaviour as follows. If `$bullets` is `TRUE`, bullets are added to every message, even when there is only one. This makes this list more predictable for vision impaired users. Note that we also added an H2 with the phrase 'Messages' to make it even easier to identify this div. This phrase can be visually suppressed via a style sheet.

Note that this routine is an exception with respect to the DIV-tags: this helper routine DOES generate its own DIVs whenever there is at least 1 message. This means that there is no DIV at all when there are no messages.

string function AdminOutput::get_html() [line 783]

construct an output page in HTML

This constructs a full HTML-page, starting at the DTD and ending with the html closing tag.

The page is constructed using nested DIVs, the layout is taken care of in a separate style sheet. All knowledge about the structure of the page is contained in this routine.

The performance of the script (# of queries, execution time) is calculated as late as possible, to catch as much as we can. Therefore the construction is done in two parts and performance is calculated last.

The contents of the various DIVs is constructed in various helper routines in order to make this routine easy to read (by humans that is). The various helper routines all are called with a string of space characters; this should improve the the readability of the page that is generated eventually.

Note that the routine `$this->get_div_messages()` does in fact generate its own DIV tags. This is done in order to completely get rid of the message DIV, we do not even want to see an empty DIV if there are no message.

string function AdminOutput::get_html_head([\$m = ""]) [line 916]

Function Parameters:

- *string* **\$m** left margin for increased readability

get all lines in the HTML head section in a single properly indented string

string function AdminOutput::get_lines(\$a, [\$m = ""]) [line 949]

Function Parameters:

- *string* **\$m** left margin for increased readability
- **\$a**

get lines from an array in a single properly indented string

This is a workhorse to convert an array of lines to a properly indented block of text.

void function AdminOutput::get_lmth() [*line 845*]

proof of concept for braille-skin

string function AdminOutput::get_logo([\$m = ""]) [*line 1052*]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct an image tag with the W

string function AdminOutput::get_menu([\$m = ""]) [*line 936*]

Function Parameters:

- *string* **\$m** left margin for increased readability

get all lines in the menu DIV in a single properly indented string

string function AdminOutput::get_navigation([\$m = ""], [\$textonly = FALSE]) [*line 1188*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *bool* **\$textonly** if TRUE, no images are used to construct navigation links

construct a navigation bar for various jobs the user is allowed to do

This constructs an unordered list (UL) with items, where each item is a clickable link to one of the various 'managers' (e.g. page manger, file manager, etc.).

There are several different 'modes' for this function. The type of output not only depends on the \$textonly flag, but also on the user's permissions for the various 'managers'. This information is retrieved from the global \$USER object.

The list of available options is stored in an array of arrays. This should make it easier to add a new 'manager' in the future: simply add another element to the \$items array below (and of course the necessary changes in the dispatcher in main_admin() above).

As a rule the links are presented to the user in the form of clickable images (icons) as provided by the currently selected skin. If the optional flag \$textonly is set, all links are displayed as a text-link (using the alt text from the image).

Depending on the user's privileges, access to some 'managers' is denied. This is visualised by displaying either a black/white image (instead of the coloured one) or by adding the class 'dimmed' to the text-based anchor tags. This makes it possible to show 'dimmed' or 'greyed out' text if the user has no access. Note, however, that these 'forbidden' links are not suppressed: the W@S philosophy says that everything should be transparent as possible and that rules out the option to suppress the things the user is not supposed to do. If the user does follow the 'dimmed' links, an error message is displayed (see the code in the dispatcher in [main_admin\(\)](#) above). One last note on this issue of denying access in a transparent way: the mousover already indicate that access will be denied. That should provide an extra clue to the user.

The list of items contains a link to the start centre (the first link) and also a link to the documentation (the last link). Both items are governed by the same privilege mask: JOB_PERMISSION_STARTCENTER. Basically it means that everyone with minimal administrator privileges has access to both the start centre and the help function. Effectively this is everyone that has access to admin.php so the images startcenter-bw.gif and help-bw.gif do not really make sense, but for completeness sake...

Note that we rely on the fact the the names of the black/white navigation images are systematically derived from the base name by appending '-bw' to the image name.

string function AdminOutput::get_pagination([\$m = ""]) [*line 1404*]

Function Parameters:

- *string* **\$m** left margin for increased readability

retrieve/construct a list of 0 or more clickable links to paginated screens

this reads the pagination-array and constructs a list of links to individual screens of a paginated display of a list.

Even when \$this->funnel_mode is TRUE, the links are still operational.

string function AdminOutput::get_popups(\$messages, [\$m = ""]) [*line 1019*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *array* **\$messages** @messages a collection of message to display via alert()

construct javascript alerts for messages

This constructs a piece of HTML that yields 0 or more calls to the javascript alert() function, once per message. If no messages need to be displayed an empty string is returned.

string function AdminOutput::get_quickbottom([\$m = ""]) [*line 1127*]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct a list of quicklinks for bottom of page

This creates HTML-code for a few links that can be displayed at the bottom of the page. Currently this list has 1 link. Possible links are

- a logout link (which will end the user's session)
- a link to the main site (i.e. /index.php without any addition parameters)
- a link to the version checker on the project's home page (see also [get_quicktop\(\)](#)).

string function AdminOutput::get_quicktop([\$m = ""]) [*line 1083*]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct a list of quicklinks for top of page, including logout link

This creates HTML-code for a few links that can be displayed at the top of the page. These links are:

- a logout link (which will end the user's session)
- a link to the main site (i.e. /index.php without any addition parameters)
- a link to the version checker on the project's home page

Note that the latter opens in a smallish new window that can be easily dismissed without 'losing' the main page.

void function AdminOutput::send_headers() [line 719]

send collected HTTP-headers to user's browser

This sends the headers that still need to be sent. These are collected in the array `$this->http_headers`. If headers are already sent, this fact is logged (and the collected headers are not sent).

void function AdminOutput::send_output() [line 748]

send collected output to user's browser

This first sends any pending HTTP-headers and subsequently outputs the page that is constructed by `$this->get_html()`. However, if the flag `suppress_output` is set then nothing is sent, not even a header. This allows for routines that interact directly with the user, e.g. a function to download a file.

bool function AdminOutput::set_funnel_mode(\$funnel_mode) [line 1698]

Function Parameters:

- *bool* **\$funnel_mode** if TRUE, funnel mode is switched on and distracting navigation links rendered inactive

manipulate the funnel mode

string function AdminOutput::set_helptopic(\$topic) [line 1544]

Function Parameters:

- *string* **\$topic** the topic in the help function

set the additional help topic to show when user clicks help button

bool function AdminOutput::set_suppress_output(\$suppress_output, \$suppress) [line 1710]

Function Parameters:

- *bool* **\$suppress** if TRUE, the actual output is suppressed (effectively disables `send_output()`)
- **\$suppress_output**

manipulate output suppression

Class AdminSkin

[line 1749]

change the looks of the user interface

This class provides 'skinned' navigation elements (icons) and styling. It is used within the [AdminOutput](#) class.

Note: We simply trust the caller to provide us with a valid filename in \$icon and \$knob (but without the .gif extension), see [get icon\(\)](#) and . This makes it a lot easier to generate the graphical icons and knobs: we simply prepend the appropriate path and append the extension .gif and there we go. The penalty for checking the actual file existence every time we generate an icon or knob is too expensive imho.

- **Package** wascore

AdminSkin::\$icon_height

int = 16 [line 1769]

- **Var** \$icon_width the vertical size of icons in this skin

AdminSkin::\$icon_path

string = [line 1763]

- **Var** \$icon_path holds the path to the graphical images used when creating icons and knobs

AdminSkin::\$icon_width

int = 16 [*line 1766*]

- **Var** \$icon_width the horizontal size of icons in this skin

AdminSkin::\$knob_height

int = 32 [*line 1775*]

- **Var** \$icon_width the vertical size of knobs in this skin

AdminSkin::\$knob_width

int = 32 [*line 1772*]

- **Var** \$icon_width the horizontal size of knobs in this skin

AdminSkin::\$name

string = [*line 1751*]

- **Var** \$name holds the name of the selected skin

AdminSkin::\$stylesheets

array = array() [*line 1760*]

- **Var** \$stylesheets contains 1 or more static stylesheets that define the skin's styling

AdminSkin::\$text_icons

bool = FALSE [line 1757]

- **Var** \$text_icons if TRUE uses 'text' rather than 'alt' parameter for text-based icons

AdminSkin::\$text_only

bool = FALSE [line 1754]

- **Var** \$text_only if TRUE limits generated icons and knobs to textual representation

Constructor *void* function AdminSkin::AdminSkin([\$name = 'base']) *[line 1782]*

Function Parameters:

- *string* **\$name** identifies the skin to setup

construct an AdminSkin object (called from AdminOutput)

string function AdminSkin::get_icon(\$icon, [\$title = "], [\$alt = "], [\$text = "]) *[line 1860]*

Function Parameters:

- *string* **\$icon** identifies the image file (without the .gif extension)
- *string* **\$title** attribute to add to graphical icon
- *string* **\$alt** attribute to add to graphical icon OR text of icon
- *string* **\$text** text of the icon if not graphical and not using alt text

return ready-to-use HTML-code for an anchor (to be used with an A-tag)

this routine can create three variations of an anchor for an icon:

- graphical: this includes the file {\$icon}.gif and the \$title and \$alt attributes
 - text with \$text: this yields a text-based icon using the \$text-parameter
 - text with \$alt: this yields a text-based icon using the \$alt-parameter
- The distinction between the latter two is made via \$this->text_icons.

string function AdminSkin::get_knob(\$knob, [\$title = "], [\$alt = "], \$icon, \$text) [*line 1900*]

Function Parameters:

- *string* **\$icon** identifies the image file (without the .gif extension)
- *string* **\$title** attribute to add to graphical icon
- *string* **\$alt** attribute to add to graphical icon OR text of icon
- *string* **\$text** text of the icon if not graphical and not using alt text
- **\$knob**

return ready-to-use HTML-code for an anchor to be used in the navigation bar

this routine can create two variations of an anchor for a knob in the navigation bar:

- graphical: this includes the file {\$icon}.gif and the \$title and \$alt attributes
- text with \$alt: this yields a text-based icon using the \$alt-parameter

The difference with the routine [get_icon\(\)](#) is that we do not have a variation with a separate \$text parameter. Another difference is that the 'knobs' in the navigation bar have different dimensions than the icons used elsewhere. (Knobs usually are 32x32 and icons are usually 16x16).

array function AdminSkin::get_stylesheets() [*line 1840*]

return the list of stylesheets associated with this skin

bool function AdminSkin::is_text_only() [*line 1831*]

is this skin a text-only skin?

Class AlertManager

[*line 55*]

Methods to access properties of an alert

This class is used to manage alerts. The following functions are supplied

- add a new alert
- edit properties of an existing alert
- delete an alert
- view list of currently existing alerts
- add a rule to an alert
- edit a rule
- delete a rule from an alert

The default action is to show a list of existing alerts.

- **Package** wascore

AlertManager::\$intervals

array = array() [line 63]

- **Var** \$intervals quick translation between cron interval and human readable values

AlertManager::\$output

object|null = NULL [line 57]

- **Var** \$output collects the html output

AlertManager::\$show_parent_menu

bool = FALSE [line 60]

- **Var** \$show_parent_menu if TRUE caller is allowed to use menu area (show config mgr menu)

Constructor *void* function AlertManager::AlertManager(&\$output) [line 73]

Function Parameters:

- *object* **&\$output** collects the html output

construct an AlertManager object

This initialises the AlertManager.

- **Uses \$CFG**

void function AlertManager::alerts_overview() [line 179]

show link to 'add an alert' followed by a list of existing alerts

this constructs an overview dialogue with options add, edit and delete alerts. Records are ordered by 'is_active' and 'full name'.

void function AlertManager::alert_delete(\$alert_id) [line 392]

Function Parameters:

- *int* **\$alert_id**

handle confirmation and actual delete of an alert

this single routine handles

- the display of a confirmation message,
- handling of the actual delete, and
- handling of the bail out/cancelbutton

void function AlertManager::alert_edit([\$alert_id = 0]) [line 239]

Function Parameters:

- **\$alert_id**

display dialogue for an alert (new or existing)

if \$alert_id is 0 [show_alert\(\)](#) presents the add alert dialogue, otherwise the edit dialogue.

void function AlertManager::alert_rule_delete(\$alert_id, \$rule_id) [line 610]

Function Parameters:

- *int* \$alert_id
- *int* \$rule_id

handle confirmation and actual delete of a rule

this single routine handles

- the display of a confirmation message,
- handling of the actual delete, and
- handling of the bail out/cancelbutton

void function AlertManager::alert_rule_edit(\$alert_id, [\$rule_id = 0]) [line 568]

Function Parameters:

- *int* \$alert_id
- *int* \$rule_id

display dialogue for a rule (new or existing)

if \$rule_id is 0 [show_rule\(\)](#) presents the add rule dialogue, otherwise the edit rule dialogue.

void function AlertManager::alert_rule_save(\$alert_id, \$rule_id) [line 705]

Function Parameters:

- *int* \$alert_id
- *int* \$rule_id

store a new or modified alert rule in the database

void function AlertManager::alert_save(\$alert_id) [line 479]

Function Parameters:

- *int* **\$alert_id**

store a new or modified alert in the database

if the user pressed cancel (or somehow no button was pressed) she is returned to the alerts overview. after that the data is validated. On error the user is returned to the same dialogue to correct errors. after adding or updating the alert the user is taken to either

- the alerts overview (after pressing [Done])
- the add rule dialogue (in case of a new alert), or
- the alert edit dialogue

This attempts to seduce the user to actually add at least one rule to the alert.

array function AlertManager::a_param(\$chore, [\$alert_id = NULL], [\$rule_id = NULL]) [line 763]

Function Parameters:

- *string* **\$chore** the next chore that could be done
- *int|null* **\$alert_id** the alert of interest or NULL if none
- *int|null* **\$rule_id** the alert rule of interest or NULL if none

shorthand for the anchor parameters that lead to the alert manager

bool function AlertManager::dialog_validate_alert(&\$dialogdef) [line 936]

Function Parameters:

- *array* **&\$dialogdef**

validate add/edit alert dialog

the usual validation and an additional check on the email address

bool function AlertManager::dialog_validate_rule(&\$dialogdef, \$alert_id, \$rule_id) [*line 959*]

Function Parameters:

- *array* &\$dialogdef
- *int* \$alert_id
- *in* \$rule_id

validate add/edit rule dialog

the usual validation and an additional check on existing rule

array function AlertManager::get_dialogdef_alert([\$alert_id = 0]) [*line 780*]

Function Parameters:

- *int* \$alert_id

construct a dialogue defintion for adding/editing selected alert properties

array function AlertManager::get_dialogdef_rule([\$alert_id = 0], [\$rule_id = 0]) [*line 849*]

Function Parameters:

- *int* \$alert_id
- *int* \$rule_id

construct a dialogdef for adding/editing a rule for alert \$alert_id

void function AlertManager::run() [*line 94*]

entry point

void function AlertManager::show_alert(&\$dialogdef, [\$alert_id = 0]) [*line 266*]

Function Parameters:

- *in* \$alert_id indicates which alert to show, 0 means add new dialogue
- &\$dialogdef

show a dialog to add a new or edit an existing alert

if \$alert_id is 0 we show the add new alert dialogue, with just three fields: Name, E-mail and Frequency (is_active is set to TRUE upon saving data). There are only two buttons: [Save] or [Cancel]. The former takes the user to the edit dialogue after adding the initial alert, the latter takes the user back to the overview.

If \$alert_id is non-0, we show the same dialogue but this time including is_active flag. This time there is also a [Done] button: this saves the changes and returns to the overview. [Save] stores the data but restarts the edit.

Also, if we are not adding but editing the current list of alert rules is displayed in a table, beneath the Save/Done/Cancel buttons. There is also an Add rule link.

bool function AlertManager::show_parent_menu() [line 162]

allow the caller to use the menu area (or not)

this routine tells the caller if it is OK to use the menu area (TRUE returned) or not (FALSE returned).

void function AlertManager::show_rule(&\$dialogdef, \$alert_id, [\$rule_id = 0]) [line 585]

Function Parameters:

- *array* **&\$dialogdef**
- *int* **\$alert_id**
- *int* **\$rule_id**

show a dialogue where a rule can be edited

we expect the caller to fill dialogdef with the correct values etc. This allows for reusing this routine to easily display errors,

void function AlertManager::tree_walk(&\$tree, &\$options, &\$area_id, &\$area_title, \$node_id) [line 899]

Function Parameters:

- *array* **&\$tree** contains all nodes in \$area_id
- *array* **&\$options** receives nodes in tree order
- *int* **&\$area_id**

- *string* **&\$area_title**
- *int* **\$node_id** start of the (sub)tree

walk the tree and add all nodes to an options array

- **Usedby** [AlertManager::tree_walk\(\)](#)
- **Uses** [AlertManager::tree_walk\(\)](#)

Class AreaManager

[line 62]

Methods to access properties of an area

This class is used to manage areas. The following functions are supplied

- add a new area (requires PERMISSION_SITE_ADD_AREA)
- set default area (requires PERMISSION_AREA_EDIT_AREA)
- delete existing area (requires PERMISSION_SITE_DROP_AREA)
- edit area properties (requires PERMISSION_AREA_EDIT_AREA)
- view list of areas (requires permissions for add, edit, delete or ACL_ROLE_INTRANET_ACCESS)

The default action is to show a list of existing areas for which the user has some form of permission. This could be either one of the permissions mentioned above or the permission to view the area (intranet).

- **Package** wascore
- **TODO** we need to take care of spurious spaces in inputs (or do we?)

AreaManager::\$areas

array = array() [*line 67*]

- **Var** list of cached area records keyed with area_id

AreaManager::\$output

object|null = NULL [*line 64*]

- **Var** collects the html output

AreaManager::\$show_parent_menu

bool = FALSE [*line 70*]

- **Var** if TRUE the calling routing is allowed to use the menu area (e.g. show config mgr menu)

Constructor *void* function AreaManager::AreaManager(&\$output) [*line 79*]

Function Parameters:

- *object* **&\$output** collects the html output

construct an AreaManager object

This initialises the AreaManager and also dispatches the chore to do.

- **Uses** \$CFG

void function AreaManager::area_add() [*line 287*]

present a dialog where the user can enter minimal properties for a new area

this displays a dialog where the user can enter the minimal necessary properties of a new area. These properties are:

- name
- public or private
- the theme to use

Other properties will be set to default values and can be edited later on, by editing the area.

The new area is saved via performing the 'chore' AREAMANAGER_CHORE_SAVE_NEW.

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER

void function AreaManager::area_delete() [line 403]

delete an area from ths site after confirmation

this either presents a confirmation dialog to the user OR deletes an area. First the user's permissions are checked and also there should be no nodes left in the area before anything is done. Only allowing deletion of an empty area is safety measure: we don't want to accidentally delete many many nodes from an area in one go (see also task_node_delete()). Also, we don't want to introduce orphaned node records (by deleting the area record without deleting nodes).

Note that this routine could have been split into two routines, with the first one displaying the confirmation dialog and the second one 'saving the changes'. However, I think it is counter-intuitive to perform a deletion of data under the name of 'saving'. So, I decided to use the same routine for both displaying the dialog and acting on the dialog.

- **TODO** since multiple tables are involved, shouldn't we use transaction/rollback/commit? Q: How well is MySQL suited for transactions? A: Mmmmm.... Which version? Which storage engine?
- **Uses** \$USER

void function AreaManager::area_edit(\$area_id) [line 710]

Function Parameters:

- *int* **\$area_id** indicates which area to edit

show the basic properties edit dialog and the edit menu

Note that this dialog does NOT allow every area property to be edited: the path to the datafiles is readonly. It feels too complicated to allow the user to actually change the path because in that case we need to move all existing files to the new location, etc. etc. I did consider to allow the GURU to perform that task (ie editing the path), but eventually decided against it: it is simply not worth it. However, if you know the way to the database and manually edit the path field you can do so, but in that case you're on your own... The user can see the path but not touch it.

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER

void function AreaManager::area_edittheme() [line 550]

show the theme/area configuration dialog and the edit menu

this displays the list of configurable properties of the theme currently associated with this area in a dialog so that the user can modify the values. Since the area-theme configuration is a more or less 'standard' list of properties, we can use the generic configuration manipulator contained in the ConfigAssistant class.

- **Uses** [ConfigAssistant](#)
- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER
- **Uses** \$CFG

void function AreaManager::area_overview() [line 198]

display list of areas with edit/delete icons etc. and option to add an area

this constructs the heart of the area manager: an optional link to add an area followed by a list of links for all areas to which the user has access. From here the user can set the default area, attempt to delete an area and edit the basic and advanced properties of an area. All actions that manipulate an area return here eventually.

Note that the calling routine (the configuration manager) is allowed to display a menu because we set the parameter `show_parent_menu` to `TRUE` here.

The constructed list looks something like this:

```

        Add an area
[H] [D] [E] (public) Exemplum Primary School (1, 10)
[ ] [D] [E] (private) Exemplum Intranet (2, 20)
[ ] [D] [E] (public) Exemplum Inactive (3, 30) (inactive)
...
```

The clickable icons [H] and [] manipulate the default area The clickable icons [D] lead to a Delete area confirmation dialog The clickable icons [E] lead to the Edit area (theme parameters) The clickable titles lead to the Edit area (basic parameters) The clickable link 'Add an area' leads to the add new area dialog.

The area titles are dimmed (grayed-out) if the user is able to see these areas (because they're public or the user has at most intranet acces for that area). Private areas for which the user has no access at all don't show up in the list. If the user has Edit-permissions, the area title is not dimmed and the area can be edited.

- **TODO** should we add a paging function to the list of areas? Currently all areas are shown in a single list...
- **TODO** should we make two categories: 'public' and 'private' in the list of areas? Maybe handy when there are many manu areas, but it would be inconsistend with the page manager menu which simply lists the areas in the sort order. Easy way out: the user is perfectly capable to set the sort order in such a way that the sort order already groups the public and private areas. Oh well....
- **Uses** `$WAS_SCRIPT_NAME`
- **Uses** `$CFG`
- **Uses** `$USER`

`void function AreaManager::area_resettheme()` [line 617]

reset the theme configuration to the factory defaults

this is a two-step process: we either show a confirmation dialog or we actually overwrite the existing theme configuration with the default values.

- **Uses** `$WAS_SCRIPT_NAME`
- **Uses** `$USER`

- **Uses** \$DB
- **Uses** \$CFG

void function AreaManager::area_save() [line 753]

validate and save modified data to database

this saves data from both the edit and the edit theme dialog if data validates. If the data does NOT validate, the edit screen is displayed again otherwise the area overview is displayed again.

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER
- **Uses** \$CFG

void function AreaManager::area_savenew() [line 933]

save the newly added area to the database

This saves the essential information of a new area to the database, using sensible defaults for the other fields. Also, a data directory is created and the relative path is stored in the new area record.

If something goes wrong, the user can redo the dialog, otherwise we return to the area overview.

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER
- **Uses** \$CFG

void function AreaManager::area_setdefault() [line 328]

make the selected area the default for the site

this sets a default area. First we check permissions and if the user

- is allowed to set the default bit on the target area, AND

- is allowed to reset the default bit on the current default area
We actually
- reset the default bit from the current default (if there is one), AND
- set the default bit for the selected node.

Note: if the user sets the default node on the current default node, the default is reset and subsequently set again (two trips to the database), This also updates the mtime of the record.

- **TODO** should we send alerts? If so, can we use the routine to queue messages from pagemanager? A reason not to send alerts: the alerts will be sent as soon as a page is added to the new area, so why bother?
- **TODO** should we acknowledge the changed default to the user or is it enough to see the icon 'move'?
- **Uses** \$USER

array function AreaManager::a_param(\$chore, [\$area_id = NULL]) [line 1531]

Function Parameters:

- *string* **\$chore** the next chore that could be done
- *int|null* **\$area_id** the area of interest or NULL if none

shorthand for the anchor parameters that lead to the area manager

int function AreaManager::count_existing_theme_properties(\$area_id, \$theme_id) [line 1634]

Function Parameters:

- *int* **\$area_id**
- *int* **\$theme_id**

determine the number of existing properties for a theme in an area

array function AreaManager::get_dialogdef_add_area() [line 1086]

construct the add area dialog

this constructs an add area dialog definition with the bare minimal fields.

array function AreaManager::get_dialogdef_delete_area() [line 1343]

construct the delete area dialog

this is basically two buttons and a CSRF token

array function AreaManager::get_dialogdef_edit_area(\$area_id) [line 1151]

Function Parameters:

- *int* **\$area_id** indicates for which area

construct the edit area basic properties dialog

Note that this dialog makes the private/public flag readonly; this field is only displayed. Also note that the datadirectory path is shown readonlye too. It is simply too much hassle to allow the user to change this path because that would imply that the existing files should move along. We'll keep it simple. However, it must be possible to look up the name of the data dir, so therefore we do display it.

void function AreaManager::get_dialog_data(&\$dialogdef, \$record) [line 1239]

Function Parameters:

- *array* **&\$dialogdef** contains dialog definition that requires the data
- *array* **\$record** conveniently holds a copy of the area record

fill the dialog with current area data from the database

Note that `area_path` is no longer a part of the dialogdef (see also [get_dialogdef_edit_area\(\)](#)) but if it were, the data would still be fetched.

string function AreaManager::get_icon_delete(\$area_id) [line 1477]

Function Parameters:

- *int* **\$area_id** the area to delete

construct a clickable icon to delete this area

- **TODO** should we check to see if the area is empty before showing delete icon? Or is it soon enough to refuse deletion when the user already clicked the icon? I'd say the latter. For now...
- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER
- **Uses** \$CFG

string function AreaManager::get_icon_edit(\$area_id) [*line 1505*]

Function Parameters:

- *int* **\$area_id** the area to edit

construct a clickable icon to edit theme properties of this area (edit advanced)

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER
- **Uses** \$CFG

string function AreaManager::get_icon_home(\$area_id, &\$areas) [*line 1421*]

Function Parameters:

- *int* **\$area_id**
- *array* **&\$areas** records with area information of all areas

construct a clickable icon to set the default area

the 'default' icon is displayed for the default area, the 'non-default' icon for all others. The user is allowed to make the area the default area if the user has edit permissions for both the old and the new default area.

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER
- **Uses** \$CFG

array function AreaManager::get_options_themes() [*line 1283*]

fetch a list of themes available for an area

this retrieves a list of themes that can be used as a list of options in a listbox or radiobuttons. Only the active themes are considered. The names of the themes that are displayed in the list are translated (retrieved from the themes language files). The list is ordered by that translated theme name.

array|bool function AreaManager::get_theme_records([\$forced = FALSE]) [*line 1324*]

Function Parameters:

- *bool* **\$forced** if TRUE forces reread from database (resets the cache)

retrieve a list of all available theme records

this returns a list of active theme-records or FALSE if none are available. The list is cached via a static variable so we don't have to go to the database more than once for this. Note that the returned array is keyed with theme_id.

TRUE function AreaManager::reset_theme_defaults(\$area_id, \$theme_id) [*line 1587*]

Function Parameters:

- *int* **\$area_id**
- *int* **\$theme_id**

reset the theme properties of an area to the default values

this deletes any existing properties for the combination of \$area_id and \$theme_id from the properties table. After that, a copy of the defaults of the theme \$theme_id is inserted in the areas-themes properties table. _Eventually_ this may exhaust the available primary keys in the area-theme-properties table. Oh well: with a handful of properties each time you need a lot of resets...

This routine returns TRUE on success or FALSE on error. In the latter case either we were not able to delete old values OR we were not able to insert a copy of the default properties.

void function AreaManager::show_edit_menu(\$area_id, [\$current_chore = ""]) [line 1368]

Function Parameters:

- *int* **\$area_id** the area currently being edited
- *string* **\$current_chore** the currently selected edit screen (used to emphasize the option in the menu)

display the edit menu via \$this->output

This displays a clickable menu on in the menu area.

bool function AreaManager::show_parent_menu() [line 140]

allow the caller to use the menu area (or not)

this routine tells the caller if it is OK to use the menu area (TRUE returned) or not (FALSE returned).

int function AreaManager::sort_order_new_area([\$at_begin = FALSE]) [line 1554]

Function Parameters:

- *bool* **\$at_begin** if TRUE the new area is placed before all others, otherwise it is added at the end

determine the value for the sort order of a new area

this calculates a new sort order value based on the existing minimum or maximum values of existing areas (if any).

Note The default sort order for areas differs from that of pages and sections: we assume that the person managing areas knows where to find the newly added area (at the bottom) whereas for a page/section maintainer it is probably more convenient to have a new page/section added at the top of the (perhaps very long) list.

Class ConfigAssistant

[line 259]

class for editing standard configuration tables

Overview

A configuration table works like this: every parameter (property, configuration item) is stored in a record in the configuration table. The core of a configuration table consists of these fields:

- name varchar(240): this is the name of the configuration parameter
- type varchar(2): parameter type: b=bool, c=checklist, d=date, dt=date/time, f=float(double), i=int, l=list, r=radio, s=string, t=time (see below for more information)
- value text: string representation of parameter value OR a comma-delimited list of values in case of a checklist
- extra text: a semicolon-delimited list of name=value pairs with additional dialog/validation information, e.g. maxlength=80 or options=true,false,filenotfound (see below for more information)
- sort_order integer: this determines the order in which parameters are presented when editing the configuration
- description text: an optional short explanation of the purpose of this parameter (in English, for internal use only)

There can be additional fields, e.g. links to parent tables in a 1-on-N relation, e.g. themes and themes_properties via theme_id. Also, the configuration table can have a separate primary key to uniquely identify a record but this is not necessary. In the config table the primary key is the name of the parameter.

Parameter types

Here is an overview of the various parameter types. - b=bool:

This type is used to store yes/no type of parameters. The parameter is considered 'TRUE' whenever the integer value of the value field is non-zero. If the integer value of the value field is zero, the parameter is considered to be 'FALSE'. Note that the NULL value also yields a zero integer value and hence 'FALSE'.

- c=checklist:

This type is an array of boolean parameters. The value of a parameter of this type is stored as a comma-delimited list of values which are to be considered 'TRUE'. If none of the elements of this array are 'TRUE', the comma-delimited list is empty. The list of possible values MUST be specified in the 'extra' field in the 'options=' item. (see below for more information about the 'extra' field).

- d=date:

This type is used to store (valid) dates, in the standard format 'yyyy-mm-dd'. (Validated in [valid_datetime\(\)](#), values from '0000-01-01' - '9999-12-31').

- dt=date/time:
This type is used to store (valid) date/time combinations, in the standard format 'yyyy-mm-dd hh:mm:ss'. (Validated in [valid_datetime\(\)](#), values from '0000-01-01 00:00:00' - '9999-12-31 23:59:59').
- f=float(double):
This type is used to store real (floating point) numbers with double precision.
- i=int:
This type is used to store integer numbers.
- l=list:
This type is used to store a single value from a list of available options (a 'picklist'). The current value is stored in the value field, and a list of possible values MUST be specified in the 'extra' field in the 'options=' item. (see below for more information about the 'extra' field).
- r=radio:
This type is also used to store a single value from a list of available options, much like the list-type (a 'picklist'). The difference is the representation in a dialog: a list parameter uses only a single line, whereas a group of radio buttons usually uses as many lines as there are available options. The current value is stored in the value field, and a list of possible values MUST be specified in the 'extra' field in the 'options=' item. (see below for more information about the 'extra' field).
- s=string:
This type is used to store generic text information. The maximum length of the string is the maximum length of the text field in the database (in MySQL this is 65535 bytes).
- t=time:
This type is used to store a (valid) time, in the format 'hh:mm:ss'. (Validated in [valid_datetime\(\)](#), values from '00:00:00' - '23:59:59').

The Extra field

This field can contain additional information about the parameter, either for validation or for display purposes. The contents of this field is a list of semicolon-delimited name=value-pairs. The following items are recognised (see also [dialoglib.php](#) - rows=<int>

This is the number of rows to display in a dialog. It can apply to string-type variables and yield a textarea-tag (as opposed to an input-tag of type 'text'). It can also apply to a set of radio buttons in a very specific way: if the number of rows is 1, the radio buttons are displayed on a single line in the dialog.

- columns=<int>
This is number of columns to use for input (but not the necessary the maximum

length of the input). If this item is omitted, default values apply, e.g. 30 for date, time, datetime; 20 for float (double) and 10 for integer, etc.

- minlength=<int>

This is the minimum number of characters that must be input. When this item is set to 1, an empty string is not allowed.

- maxlength=<int>

This is the maximum number of characters that can be input.

- minvalue=<mixed>

This is the minimum value for the field. The type of the minimum value is the same as the type of the variable itself. It applies to integers, floats, dates, datetimes and times.

- maxvalue=<mixed>

This is the maximum value for the field. The type of the maximum value is the same as the type of the variable itself. It applies to integers, floats, dates, datetimes and times.

- decimals=<int>

This is the number of decimals that should be displayed in dialogs. It applies to floats.

- options=option1,option2,option3,...)

This is a comma-delimited list of valid values for a list, radio or checklist parameter. The value of the parameter is one of the options in this list (for parameter types list and radio) OR a comma-delimited list of zero or more items (for checklists).

- viewonly=<int>

If the integer value of this item is non-zero, the user is not allowed to edit the value of the parameter. However, it is supposed to be displayed in the dialog nevertheless.

Generating dialogs for editing

The ConfigAssistant is clever enough to read and write the configuration parameters from the specified table, using a where-clause when necessary. Also, the ConfigAssistant automatically constructs translations of screen prompts in a very specific way when constructing dialogs.

The translation keys are constructed as follows. - simple string-like parameters (string, date, time, etc.)

name = <prefix><name>

label = <prefix><name>_label

title = <prefix><name>_title

- bool parameter
name = <prefix><name>
label = <prefix><name>_label
title = <prefix><name>_title
option = <prefix><name>_option

- list or radio parameter
name = <prefix><name>
label = <prefix><name>_label,
title = <prefix><name>_title
options:
label = <prefix><name>_<option1>_label
title = <prefix><name>_<option1>_title
...
label = <prefix><name>_<optionN>_label
title = <prefix><name>_<optionN>_title

- checklist parameter
name = <prefix><name>
label = <prefix><name>_label
title = <prefix><name>_title
options:
title = <prefix><name>_<option1>_title
option = <prefix><name>_<option1>_option
...
title = <prefix><name>_<optionN>_title
option = <prefix><name>_<optionN>_option

The string <prefix> can be used to avoid name clashes in the 'admin' (or other) language file. The translations are then looked up in the specified language domain (default: admin).

Examples

Example 1: sending a dialog to the user for editing all the parameters in the the main config table:

```
$table = 'config';
$keyfield = 'name';
$assistant = new ConfigAssistant($table,$keyfield);
$href = 'index.php?job=(...)&task=editconfig';
$assistant->show_dialog($output,$href);
```

Example 2: saving the modified data to the table

```
$table = 'config';
$keyfield = 'name';
$assistant = new ConfigAssistant($table,$keyfield);
if (!$assistant->save_data($output)) {
```

```

    echo "FAILED";
} else {
    echo "SUCCESS saving configuration";
}

```

Sounds easy to use, doesn't it?

- **Package** wascore
- **TODO** implement checklist
- **Usedby** [AreaManager::area_edittheme\(\)](#)

ConfigAssistant::\$dialogdef

array = NULL [line 282]

- **Var** \$dialogdef an array with a dialog ready to use for [dialog_quickform\(\)](#)

ConfigAssistant::\$dialogdef_hidden

array = [line 285]

- **Var** \$dialogdef_hidden array with additional fields that should be included in the dialog

ConfigAssistant::\$fields

array = array('name','type','value','extra') [line 264]

- **Var** \$fields the list of essential fields to retrieve from the the table

ConfigAssistant::\$keyfield

string = [line 270]

- **Var** \$keyfield a string indicating the keyfield to uniquely identify the configuration parameter

ConfigAssistant::\$language_domain

\$string = [line 279]

- **Var** \$domain the language domain where to look for translations (default: 'admin')

ConfigAssistant::\$prefix

\$string = [line 276]

- **Var** \$prefix is prepended for every translation/language key and the dialog item name

ConfigAssistant::\$records

array = NULL [line 273]

- **Var** \$records the cached list of configuration values straight from the database

ConfigAssistant::\$table

string = [line 261]

- **Var** \$table the table that contains the configuration

ConfigAssistant::\$where

mixed = [line 267]

- **Var \$where** a string with a whereclause (without 'WHERE') or an array with conditions

Constructor *void* function ConfigAssistant::ConfigAssistant(\$table, \$keyfield, [\$prefix = ''], [\$domain = ''], [\$where = ''], [\$dialogdef_hidden = '']) [line 300]

Function Parameters:

- *string* **\$table** the table where the configuration parameters are stored
- *string* **\$keyfield** the field that uniquely identifies the configuration parameters
- *string* **\$prefix** is prepended for every translation/language key and the also dialog item name
- *string* **\$domain** the language domain where to look for translations (default: 'admin')
- *mixed* **\$where** a whereclause (without 'WHERE') or an array with conditions
- *array* **\$dialogdef_hidden** additional fields for inclusion in dialog definition

constructor for the configuration assistant

This stores the parameters, sets defaults when applicable and subsequently reads selected config parameters into the \$this->records for future reference.

array function ConfigAssistant::get_dialogdef() [line 391]

construct an array with the dialog information

- **TODO** implement checklist

array function ConfigAssistant::get_extra(\$type, \$extras) [line 505]

Function Parameters:

- *string* **\$type** variable type (necessary for calculating minvalue/maxvalue)

- *string* **\$extras** semicolon-delimited list of name=value pairs

construct an array based on name=value pairs in an 'extra' field

This constructs an array based on the name=value pairs in the extras string. Most recognised parameters yield an integer value. Exceptions are: minvalue and maxvalue: these yield a variable of the same type as the config parameter itself options yields an array with all options from the comma delimited list Unknown name=value pairs are logged with WLOG_DEBUG.

void function ConfigAssistant::get_options_from_extra(\$extra, \$name) [line 547]

Function Parameters:

- **\$extra**
- **\$name**

void function ConfigAssistant::save_data(&\$output) [line 337]

Function Parameters:

- *object* **&\$output** the object that collects the output

save the modified configuration parameters to the database

- Uses [dialog_validate\(\)](#)

void function ConfigAssistant::show_dialog(&\$output, \$href) [line 324]

Function Parameters:

- *object* **&\$output** the object that collects the output
- *string* **\$href** the target for the form that will be created

add a complete dialog to the content area of the output

- Uses [dialog_quickform\(\)](#)

Class DatabaseMysql

[line 58]

MySQL database

This implements access to the MySQL database.

- **Package** wascore

DatabaseMysql::\$charset

mixed = NULL [line 102]

- **Var** \$charset is the MySQL character set to use (e.g. utf8 or utf8mb4)

DatabaseMysql::\$collation

mixed = NULL [line 105]

- **Var** \$collation is the MySQL collation to use (e.g. utf8_unicode_ci or utf8mb4_unicode_ci)

DatabaseMysql::\$db_link

resource = [line 78]

- **Var** \$db_link is the database link identifier

DatabaseMysql::\$db_name

string = [line 72]

- **Var** \$dn_name is the name of the database to use, e.g. 'was'

DatabaseMysql::\$db_password

string = [line 69]

- **Var** \$db_password is part of credentials for database access

DatabaseMysql::\$db_server

string = [line 63]

- **Var** \$db_server is the name of the database server, e.g. 'localhost' or 'db.example.com:3306'

DatabaseMysql::\$db_type

string = [line 60]

- **Var** \$db_type is the database type, always 'mysql' for this class

DatabaseMysql::\$db_username

string = [line 66]

- **Var** \$db_username is part of credentials for database access

DatabaseMysql::\$db_version

string = [line 81]

- **Var** \$db_version is the database version string. e.g. '5.1.32' or '4.0.23a' or FALSE if unavailable

DatabaseMysql::\$debug

bool = [line 93]

- **Var** \$debug if TRUE switch debugging on

DatabaseMysql::\$engine

mixed = NULL [line 99]

- **Var** \$engine is the MySQL engine to use (e.g. MyISAM or InnoDB)

DatabaseMysql::\$errno

integer = [line 87]

- **Var** \$errno is the error number generated by the latest mysql command

DatabaseMysql::\$error

string = [line 90]

- **Var \$error** is the error message generated by the latest mysql command

DatabaseMysql::\$prefix

string = [line 75]

- **Var \$prefix** is the table name prefix, e.g. 'was_'

DatabaseMysql::\$query_counter

integer = [line 84]

- **Var \$query_counter** is the number of queries executed so far

DatabaseMysql::\$utf8_support

int = [line 96]

- **Var \$utf8_support** is the level of UTF-8 support: (none), 3 (limited), 4 (full, but with a quirky name)

Constructor *void* function DatabaseMysql::DatabaseMysql(\$prefix, [\$debug = FALSE]) [line 114]

Function Parameters:

- *string* **\$prefix** table name prefix, e.g. 'was_'
- *bool* **\$debug** if TRUE extra information is displayed (handy for debugging the code)

initialise query counter and other variables, store the table prefix

void function DatabaseMysql::check_engine() [line 986]

determine the database engine (and charset and collation) to use for new tables

this routine attempts to find the engine that is/was used to create existing tables with the prefix for this installation. As a side effect we also try to determine the charset and the collation. This is all based on the output of `SHOW TABLE STATUS LIKE '{$prefix}%'`;

If we get an empty result set, it means there is no table yet from which we can learn the engine/charset/collation. In that case we want to use the database default engine with an appropriate charset/collation. We indicate this to the caller by returning `FALSE`.

If, however, we have at least one existing table with prefix `$prefix`, the `SHOW TABLE STATUS` query returns at least 1 result row. From this row we can use the column 'Engine' to find out which engine was used for that table. From the column 'Collation' we can extract the collation used. Once we have that, we can also (re-)construct the charset (because a collation is always based on the charset, e.g. 'utf8' and 'utf8_unicode_ci') by looking at the string upto the first '_'.

If the `SHOW TABLE STATUS` query does return a result but there is no field 'Engine' and possibly also no field 'Collation', we simply leave those parameters blank. This is based on the assumption that IF the field 'Engine' does not exist, we are dealing with a very old version of MySQL < 4.1.1 which has no support for utf8 anyway.

So, usage of this routine in `create_table_sql()` is something like this: ...

```
if ($this->check_engine()) {
    $sql .= (empty($this->engine)) ? " : ' ENGINE='{$this->engine}';
    $sql .= (empty($this->charset)) ? " : ' DEFAULT CHARSET='{$this->charset}';
    $sql .= (empty($this->collation)) ? " : ' COLLATE='{$this->collation}';
} else {
    switch($this->utf8_support) {
        case 3: $sql .= ' DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci'; break;
        case 4: $sql .= ' DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci'; break;
    }
}
...
```

The reason to go through all this trouble is as follows. Suppose we have installed W@S in a database 5.1.0 with MyISAM as the default engine and an appropriate charset and collation (utf8_unicode_ci). That implies that all core tables are MyISAM / utf8 / utf8_unicode_ci. OK. Now this database is moved to a newer version, say 5.5.39, with InnoDB as default. Now add a module with a new table with a foreign key on one of the core tables, say `was_users`. Boom! Error 150 and no table is created. Aaarghhh.

This heuristic workaround is to look at existing tables in the database with 'our' prefix and copy the engine/charset/collation from there. That means that at least all tables, old and new, have the same engine, etc. (The same logic applies to a migration from a pre 5.5.3 database with only utf8 to a newer one where perhaps utf8mb4 would be default.

For a new installation it doesn't matter: the first table that gets installed is installed with the default engine. The second table looks up the properties of the first (in this routine `check_engine()`) and uses those. The answers are cached in the `$DB`-object so we only have to do this twice for a new installation: the first time we leave here in step 3 with `FALSE`, the second time we leave here in step 5 with `TRUE` and after that we return immediately with `TRUE` In step 1.

I must say that this is all pretty confusing and not straightforward (more like straightbackward...).

Note that this can still blow up if there is a real mysql-error in steps 2/3/4 because that implies to use the database's default values which may have changed since the original installation. Oh well. The chances are slim. I hope. I wish.

bool function DatabaseMysql::close() [line 180]

close the connection to the database

this closes the connection and resets the resource `$this->db__link`, in order to prevent use of the connection after it is already closed.

bool/string function DatabaseMysql::column_definition(\$fielddef) [line 602]

Function Parameters:

- *array \$fielddef* an array that describes a field in a generic way

convert a fielddef array to a MySQL specific column definition

this creates a MySQL specific column definition based on a generic field description. The following keywords are recognised:

- *name*: the name of the field (required)
- *type*: the generic type of the field (see table below for supported types) (required)
- *length*: the size of a numeric field or the length of a textual field
- *decimals*: the number of decimals in real numbers (implies length)
- *unsigned*: used for numeric fields that don't need negative values (see note 5 below)
- *notnull*: if true, the field is not allowed/able to contain the `NULL` value
- *default*: the default value for the field
- *enum_values*: an array with allowable values for the enum field
- *comment*: a string that can be used to document the field definition

The generic field types are mapped to actual MySQL data types via a lookup table. Most types have a number of boolean flags which indicate how the field definition must be interpreted. Entries with an empty type are considered special cases.

- *len* = 1: look for a length parameter and use it if it is defined.
- *dec* = 1: look for a decimals parameter and use it if it is defined. Implies *len*.

- unsigned = 1: look for an unsigned parameter and use it if it is defined and true (see note 5 below).
- default = 1: look for a default parameter, and use it if it is defined.
- quote = 1: if a default is allowed AND defined, use single quotes + escaped string.

Note 1: a 1 in the table means that the field type `_allows_` the corresponding parameter, and 0 means that this parameter is NOT allowed. It doesn't say anything about parameters being `_required_` (e.g. a varchar must have a length). It is the responsibility of the author of the field definition to provide all necessary parameters.

Note 2: `enum_values` are not used at this time; an enum-field simply maps to a varchar field and that's it. Adding the `enum_values` to a data definition does help to document the purpose of the field. The reason for not (yet) implementing enums in full is the issues associated with the translations in the UI. Furthermore, using native enums in MySQL is a royal PITA. For now the application should know what it is doing when using and updating enums in a table.

Note 3: at this time it is not possible to set the default value of a text-type field to the string consisting of the letters N, U, L and L: that caseinsensitive string is always interpreted as the NULL-value in the database, ie. it yields "DEFAULT NULL" and not "DEFAULT 'NULL'" or "DEFAULT 'Null'" or "DEFAULT 'null'".

Note 4: even though comments may or may not be stored in the MySQL database, the corresponding COMMENT-clauses are generated, if only for documentation/debugging purposes. These clauses are correctly parsed by MySQL but they are subsequently discarded in old MySQL-versions.

Note 5: As of version 2009051401 (0.0.5) the parameter 'unsigned' is deprecated. This is because even though MySQL implements an unsigned attribute for numeric fields, other RDBMSs might not. Therefore we stay away from this MySQL-specific construct as much as possible. Also, the translation from the 'serial' fieldtype no longer adds the unsigned attribute to the actual MySQL definition.

- **TODO** should we allow both int and integer?
- **TODO** 'enum' type equivalent with varchar, `enum_values[]` array is not used at all, only as a form of documentation

concatenation function DatabaseMysql::concat(\$string1, \$string2) [line 283]

Function Parameters:

- *string* **\$string1** contains a quoted/escaped string, a fieldname or other expression
- *string* **\$string2** contains a quoted/escaped string, a fieldname or other expression

helper function for string concatenation in sql statements

From <http://troels.arvin.dk/db/rdbms/#functions-concat>:

SQL Standard: Core feature ID E021-07: Concatenating two strings is done with the || operator:

string1 || string2

If at least one operand is NULL, then the result is NULL.

MySQL: Badly breaks the standard by redefining || to mean OR. Offers instead a function, CONCAT(string, string), which accepts two or more arguments.

In order to try and stay as database-independent as possible without losing this concatenation feature, we have to resort to a database-specific function. Aaargggghh!

Typical use of this function:

```
...
$sql = 'UPDATE {'$DB->prefix}table ' .
      'SET message = ' . $DB->concat('message', "'". $DB->escape("addition to
message\n")."'").' ' .
      'WHERE table_id = ' . $some_id;
$DB->exec($sql);
...
```

and that's exactly the kind of hairy code I'd like to stay away from, because of the necessary delicate balancing of quotes and the amount of thought it requires to get a query right. Obviously it is much easier (and less quote-error prone and almost elegant) to do something like this:

```
$record = db_select('tablename', 'message', array('table_id' => $some_id));
$value = $record['message'] . "addition to message\n";
db_update('tablename', array('message' => $value), array('table_id' => $some_id));
```

Note that 'Standard-SQL' would yield almost as much trouble with concatenation, even if it were possible to use in MySQL, e.g.:

```
...
$sql = 'UPDATE {'$DB->prefix}table ' .
      "SET message = message || ' " . $DB->escape("addition to message\n")." ' .
      'WHERE table_id = ' . $some_id;
$DB->exec($sql);
...
```

No matter what: this is ugly. The reason I still want to use this kind of code is that (much) more expensive to use series of SELECT / UPDATE statements with concatenation in PHP, not to mention the fact that using two separate SQL-statements introduces race conditions, so there. Alas this is database specific.

Note that some of the quote-hell can be dealt with via [db_escape_and_quote\(\)](#):

```
$sql = 'UPDATE {'$DB->prefix}table ' .
      'SET message = ' . $DB->concat('message', db_escape_and_quote("addition to
message\n")).' ' .
      'WHERE table_id = ' . $some_id;
$DB->exec($sql);
```

- **TODO** perhaps extend this function to accept more than 2 strings?

bool function DatabaseMysql::connect(\$db_server, \$db_username, \$db_password, \$db_name) [*line 142*]

Function Parameters:

- *string* **\$db_server** database server, e.g. 'localhost' or 'db.example.com:3306'
- *string* **\$db_username** part of credentials
- *string* **\$db_password** part of credentials
- *string* **\$db_name** database to use, e.g. 'was'

connect to the database server and open the database

this opens a connection to the database, perhaps sets some connection parameters and subsequently selects the requested database. If anything goes wrong, FALSE is returned and additional information can be retrieved from the variables \$this->errno and \$this->error.

Note that for 4.1.x <= MySQL < 5.5.3 we use charset utf8 for the connection, whereas for 5.5.3 and up we use the utf8mb4 charset which can handle UTF-8 with upto 4-byte sequences.

- **TODO** weigh pros and cons of persistent database connections, perhaps add as config option?

int|bool function DatabaseMysql::create_table(\$tabledef) [*line 448*]

Function Parameters:

- *array* **\$tabledef** a generic table definition (not database-specific)

create a table via a generic (non-MySQL-specific) table definition

this executes a MySQL-specific CREATE TABLE statement based on a generic table definition. The actual work is done in create_table_sql(), which makes it possible to see the result of converting a generic definition to an actual table; very handy while debugging.

- **TODO** document correct link for documentation of generic table definition 'tabledefs.php'

string/bool function DatabaseMysql::create_table_sql(\$tabledef) [*line 475*]

Function Parameters:

- *array* **\$tabledef** a generic table definition (not database-specific)

create the MySQL-specific SQL statement to create a table via a generic table definition
 this creates a MySQL-specific CREATE TABLE statement from a generic table definition.
 See [tabledefs.php](#) for more information about the format of this generic table definition.

Note that this routine takes the level of UTF-8-support into account; for level 3 we use charset 'utf8' and for level 4 'utf8mb4' (see also [connect\(\)](#)).

The foreign key constraint is now capable of naming the constraints with a unique (database wide) symbol. (Cures InnoDB-error 1005 (HY000) errno: 121). This symbol is the prefixed tablename followed by either the specified key name or a 1-based integer uniquemaker per table.

- **TODO** document correct link for documentation of generic table definition 'tabledefs.php'
- **TODO** find a way to deal with the enum values: where do we keep them? Or do we keep them at all?

int/bool function DatabaseMysql::drop_table(\$tablename) [*line 412*]

Function Parameters:

- *string* **\$tablename** the name of the table to drop (prefix will be added automatically)

unconditionally drop the specified table

bool function DatabaseMysql::dump(&\$data, [\$drop = TRUE], [\$tables = ""]) [*line 729*]

Function Parameters:

- *string* **&\$data** receives the dump of the tables
- *bool* **\$drop** if TRUE add code to drop the table before the data definition
- *mixed* **\$tables** array with names of tables to dump, empty (string, array) means all our tables

make a text dump of our tables in the database suitable for backup purposes

This creates a text dump of the selected tables in parameter \$data. If \$tables is empty we dump all the tables in the database that start with 'our' prefix (there could be other websites using the same table with another prefix, we won't dump those). If \$tables is an array, it is assumed to be an array with table names without our prefix. In this case we will prepend the prefix. If parameter \$drop is TRUE, we add code to drop the table from the database (if it exists) before we recreate it.

If there were no errors, we return TRUE (and \$data contains the dump). If errors were encountered, we return FALSE and \$this->errno and \$this->error can tell the caller what happened. If there were errors, \$data is undefined.

Strategy is as follows. First we make a valid list of tables to dump, either by asking the database for a list of tables LIKE "{\$prefix}%" or by manipulating and validating the array \$tables that was provided by the caller.

Subsequently we let the database generate a CREATE TABLE statement and then we step through the data (if any) and add it to \$data.

Note MySQL is quite liberal in what it accepts as field values. However, I try to generate INSERT INTO-statements as clean as possible by NOT quoting numeric values. HTH. It still is a MySQL-specific dump, though. You cannot simply use the result 'as-is' to migrate to another database.

string/bool function DatabaseMysql::escape(\$unesaped_string) [line 206]

Function Parameters:

- *string* **\$unesaped_string** the string to escape

escape special characters in string

this makes sure that dangerous characters like single quotes are properly escaped. this routine has a special twist because of the limited support for UTF-8 in some MySQL-versions. If support is limited (\$this->utf_support equals 3), we strip any 4-byte UTF-8 characters from the string and we replace those with the generic substitution character U+FFFD (see als [mysql utf8mb3](#)).

The (bold) assumptions here are:

1. the \$unesaped_string is in fact proper UTF-8 (see [utf8_validate\(\)](#)), and
2. all strings that are headed for the database are funneled through this routine, always.

int|bool function DatabaseMysql::exec(\$sql) [*line 301*]

Function Parameters:

- *string* **\$sql** valid SQL statement

execute an action query and return the number of affected rows

this method should be used to execute SQL-statements that do NOT return a result set, use query() for that. This method works well for INSERTs, DELETEs and UPDATEs. If there is an error this method returns FALSE. Otherwise it returns the number of affected lines. Note that there is a difference between 0 affected lines and FALSE.

int|bool function DatabaseMysql::last_insert_id([\$table_name = "], [\$field_name = "]) [*line 349*]

Function Parameters:

- *string* **\$table_name** (optional) tablename (unused in MySQL)
- *string* **\$field_name** (optional) fieldname (unused in MySQL)

retrieve the most recent automatically inserted id ('auto_increment')

this method returns the id that was automatically generated in the previous INSERT-query or 0 if the previous query was not an INSERT query.

Note: as per the MySQL manual this function returns an int and not a bigint. If the auto_increment field is a bigint, this method returns an incorrect result. There is a work-around (e.g. SELECT LAST_INSERT_ID()) but this is not the way it works in Website@School; all id's are simple int's and not bigint's, so there is no need to generate yet another query after every insert.

Note that this method can be called with a table name and a field name. This is a hook for future expansion; this MySQL-driver does not actually use it. However, it is handy to always call this method with table and field name to make adding a new database driver easier.

string function DatabaseMysql::mysql_utf8mb3(\$utf8str) [*line 905*]

Function Parameters:

- *string* **\$utf8str** valid UTF-8 encoded string

message string to contain only 3-byte UTF8-sequences

this replaces an otherwise perfectly valid 4-byte UTF-8 sequence in \$utf8str with a 3-byte UTF-8 sequence equivalent with the Unicode replacement character U+FFFD.

The effect is that it leaves a hint that there used to be some character instead of silently discarding 4-byte sequences which MySQL does.

int/bool function DatabaseMysql::mysql_utf8_support(\$db_link) [*line 875*]

Function Parameters:

- *resource* **\$db_link** the MySQL connection

determine the level of UTF-8 support based on MySQL-server version

MySQL support for UTF-8 was non-existent before 4.1.x and limited until 5.5.3. In this context 'limited' means: only the Basic Multilingual Plane (U+0000 ... U+FFFF) is supported, i.e. a maximum of 3-byte sequences per character.

As of 5.5.3 the full UTF-8 specification according to RFC 3629 is implemented. MySQL now has 'invented' yet another proprietary name for this character set: 'utf8mb4' (WTF?), and introduces the alias 'utf8mb3' for the pre 5.5.3 limited support for 'utf8' (WTF??), hinting that the meaning of 'utf8' may change in future versions to indicate 'utf8mb4' (WTF???). IM(NS)HO this is yet another reason to go looking for a decent replacement for MySQL. YMMV.

This routine returns exactly one of the values below (based on the server version).

- 0: there is no UTF-8 support available in this server
- 3: the limited 3-byte sequences are supported
- 4: full support for 4-byte sequences available, but using the stupid ad-hoc name 'utf8mb4' or the value FALSE if version information could not be obtained.

As a side effect, we record the server version information in \$this->db_version (which is handy when creating a backup and also for debugging purposes).

object/bool function DatabaseMysql::query(\$sql, [\$limit = "], [\$offset = "]) [*line 380*]

Function Parameters:

- *string* **\$sql** valid SQL statement
- *int* **\$limit** optional limitation of the number of records returned
- *int* **\$offset** optional number of records to skip; \$offset = 0 means start with the first

execute a select query and return a result set

this method should be used to execute SQL-statements that do return a result set: SELECT, SHOW, EXPLAIN and DESCRIBE. Use exec() for action queries . If there is an error this method returns FALSE. Otherwise it returns a result set in the form of a DatabaseMysqlResult object.

if parameters \$limit and \$offset are set the result set contains at most \$limit rows, starting at offset \$offset. it is OK to specify just \$limit; \$offset is taken into account only when \$limit is specified too. Note that different databases have a different syntax for limiting the number of returned records. MySQL supports both 'LIMIT limit OFFSET offset' (which we use here) and 'LIMIT offset,limit'.

the LIMIT-clause is blindly appended to the SQL-statement; it is up to the caller to decide wheter specifying a limit and an offset make sense or not.

bool function DatabaseMysql::table_exists(\$tablename) [*line 423*]

Function Parameters:

- *string* **\$tablename** name of the table to check (prefix will be added automatically)

see if the named table exists

Class DatabaseMysqli

[*line 60*]

MySQL database

This implements access to the MySQL database.

- **Package** wascore

DatabaseMysqli::\$charset

mixed = NULL [line 104]

- **Var** \$charset is the MySQL character set to use (e.g. utf8 or utf8mb4)

DatabaseMysqli::\$collation

mixed = NULL [line 107]

- **Var** \$collation is the MySQL collation to use (e.g. utf8_unicode_ci or utf8mb4_unicode_ci)

DatabaseMysqli::\$db_link

resource = [line 80]

- **Var** \$db_link is the database link identifier

DatabaseMysqli::\$db_name

string = [line 74]

- **Var** \$db_name is the name of the database to use, e.g. 'was'

DatabaseMysqli::\$db_password

string = [line 71]

- **Var** \$db_password is part of credentials for database access

DatabaseMysqli::\$db_server

string = [line 65]

- **Var** \$db_server is the name of the database server, e.g. 'localhost' or 'db.example.com:3306'

DatabaseMysqli::\$db_type

string = [line 62]

- **Var** \$db_type is the database type, always 'mysqli' for this class

DatabaseMysqli::\$db_username

string = [line 68]

- **Var** \$db_username is part of credentials for database access

DatabaseMysqli::\$db_version

string = [line 83]

- **Var** \$db_version is the database version string. e.g. '5.1.32' or '4.0.23a' or FALSE if unavailable

DatabaseMysqli::\$debug

bool = [line 95]

- **Var** \$debug if TRUE switch debugging on

DatabaseMysqli::\$engine

mixed = NULL [line 101]

- **Var** \$engine is the MySQL engine to use (e.g. MyISAM or InnoDB)

DatabaseMysqli::\$errno

integer = [line 89]

- **Var** \$errno is the error number generated by the latest mysqli command

DatabaseMysqli::\$error

string = [line 92]

- **Var** \$error is the error message generated by the latest mysqli command

DatabaseMysqli::\$prefix

string = [line 77]

- **Var** \$prefix is the table name prefix, e.g. 'was_'

DatabaseMysqli::\$query_counter

integer = [line 86]

- **Var** \$query_counter is the number of queries executed so far

DatabaseMysqli::\$utf8_support

int = [line 98]

- **Var** \$utf8_support is the level of UTF-8 support: (none), 3 (limited), 4 (full, but with a quirky name)

Constructor *void* function DatabaseMysqli::DatabaseMysqli(\$prefix, [\$debug = FALSE]) *[line 116]*

Function Parameters:

- *string* **\$prefix** table name prefix, e.g. 'was_'
- *bool* **\$debug** if TRUE extra information is displayed (handy for debugging the code)

initialise query counter and other variables, store the table prefix

void function DatabaseMysqli::check_engine() *[line 1005]*

determine the database engine (and charset and collation) to use for new tables

this routine attempts to find the engine that is/was used to create existing tables with the prefix for this installation. As a side effect we also try to determine the charset and the collation. This is all based on the output of SHOW TABLE STATUS LIKE '{\$prefix}%';

If we get an empty result set, it means there is no table yet from which we can learn the engine/charset/collation. In that case we want to use the database default engine with an appropriate charset/collation. We indicate this to the caller by returning FALSE.

If, however, we have at least one existing table with prefix \$prefix, the SHOW TABLE STATUS query returns at least 1 result row. From this row we can use the column 'Engine' to find out which engine was used for that table. From the column 'Collation' we can extract the collation used. Once we have that, we can also (re-)construct the charset (because a collation is always based on the charset, e.g. 'utf8' and 'utf8_unicode_ci') by looking at the string upto the first '_'.

If the SHOW TABLE STATUS query does return a result but there is no field 'Engine' and possibly also no field 'Collation', we simply leave those parameters blank. This is based on the assumption that IF the field 'Engine' does not exist, we are dealing with a very old version

of MySQL < 4.1.1 which has no support for utf8 anyway.

So, usage of this routine in `create_table_sql()` is something like this: ...

```
if ($this->check_engine()) {
    $sql .= (empty($this->engine)) ? " : ' ENGINE='.$this->engine;
    $sql .= (empty($this->charset)) ? " : ' DEFAULT CHARSET='.$this->charset;
    $sql .= (empty($this->collation)) ? " : ' COLLATE='.$this->collation;
} else {
    switch($this->utf8_support) {
        case 3: $sql .= ' DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci'; break;
        case 4: $sql .= ' DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci'; break;
    }
}
...
```

The reason to go through all this trouble is as follows. Suppose we have installed W@S in a database 5.1.0 with MyISAM as the default engine and an appropriate charset and collation (`utf8_unicode_ci`). That implies that all core tables are MyISAM / utf8 / `utf8_unicode_ci`. OK. Now this database is moved to a newer version, say 5.5.39, with InnoDB as default. Now add a module with a new table with a foreign key on one of the core tables, say `was_users`. Boom! Error 150 and no table is created. Aaarghhh.

This heuristic workaround is to look at existing tables in the database with 'our' prefix and copy the engine/charset/collation from there. That means that at least all tables, old and new, have the same engine, etc. (The same logic applies to a migration from a pre 5.5.3 database with only utf8 to a newer one where perhaps utf8mb4 would be default.

For a new installation it doesn't matter: the first table that gets installed is installed with the default engine. The second table looks up the properties of the first (in this routine `check_engine()`) and uses those. The answers are cached in the `$DB`-object so we only have to do this twice for a new installation: the first time we leave here in step 3 with FALSE, the second time we leave here in step 5 with TRUE and after that we return immediately with TRUE In step 1.

I must say that this is all pretty confusing and not straightforward (more like straightbackward...).

Note that this can still blow up if there is a real mysql-error in steps 2/3/4 because that implies to use the database's default values which may have changed since the original installation. Oh well. The chances are slim. I hope. I wish.

bool function DatabaseMysqli::close() [line 196]

close the connection to the database

this closes the connection and resets the resource `$this->db__link`, in order to prevent use of the connection after it is already closed.

Function Parameters:

- array **\$fielddef** an array that describes a field in a generic way

convert a fielddef array to a MySQL specific column definition

this creates a MySQL specific column definition based on a generic field description. The following keywords are recognised:

- name: the name of the field (required)
- type: the generic type of the field (see table below for supported types) (required)
- length: the size of a numeric field or the length of a textual field
- decimals: the number of decimals in real numbers (implies length)
- unsigned: used for numeric fields that don't need negative values (see note 5 below)
- notnull: if true, the field is not allowed/able to contain the NULL value
- default: the default value for the field
- enum_values: an array with allowable values for the enum field
- comment: a string that can be used to document the field definition

The generic field types are mapped to actual MySQL data types via a lookup table. Most types have a number of boolean flags which indicate how the field definition must be interpreted. Entries with an empty type are considered special cases.

- len = 1: look for a length parameter and use it if it is defined.
- dec = 1: look for a decimals parameter and use it if it is defined. Implies len.
- unsigned = 1: look for an unsigned parameter and use it if it is defined and true (see note 5 below).
- default = 1: look for a default parameter, and use it if it is defined.
- quote = 1: if a default is allowed AND defined, use single quotes + escaped string.

Note 1: a 1 in the table means that the field type `_allows_` the corresponding parameter, and 0 means that this parameter is NOT allowed. It doesn't say anything about parameters being `_required_` (e.g. a varchar must have a length). It is the responsibility of the author of the field definition to provide all necessary parameters.

Note 2: `enum_values` are not used at this time; an enum-field simply maps to a varchar field and that's it. Adding the `enum_values` to a data definition does help to document the purpose of the field. The reason for not (yet) implementing enums in full is the issues associated with the translations in the UI. Furthermore, using native enums in MySQL is a royal PITA. For now the application should know what it is doing when using and updating enums in a table.

Note 3: at this time it is not possible to set the default value of a text-type field to the string consisting of the letters N, U, L and L: that caseinsensitive string is always interpreted as the NULL-value in the database, ie. it yields "DEFAULT NULL" and not "DEFAULT 'NULL'" or "DEFAULT 'Null'" or "DEFAULT 'null'".

Note 4: even though comments may or may not be stored in the MySQL database, the corresponding COMMENT-clauses are generated, if only for documentation/debugging

purposes. These clauses are correctly parsed by MySQL but they are subsequently discarded in old MySQL-versions.

Note 5: As of version 2009051401 (0.0.5) the parameter 'unsigned' is deprecated. This is because even though MySQL implements an unsigned attribute for numeric fields, other RDBMSs might not. Therefore we stay away from this MySQL-specific construct as much as possible. Also, the translation from the 'serial' fieldtype no longer adds the unsigned attribute to the actual MySQL definition.

- **TODO** should we allow both int and integer?
- **TODO** 'enum' type equivalent with varchar, enum_values[] array is not used at all, only as a form of documentation

concatenation function DatabaseMysqli::concat(\$string1, \$string2) [line 299]

Function Parameters:

- *string \$string1* contains a quoted/escaped string, a fieldname or other expression
- *string \$string2* contains a quoted/escaped string, a fieldname or other expression

helper function for string concatenation in sql statements

From <http://troels.arvin.dk/db/rdbms/#functions-concat>:

SQL Standard: Core feature ID E021-07: Concatenating two strings is done with the || operator:

string1 || string2

If at least one operand is NULL, then the result is NULL.

MySQL: Badly breaks the standard by redefining || to mean OR. Offers instead a function, CONCAT(string, string), which accepts two or more arguments.

In order to try and stay as database-independent as possible without losing this concatenation feature, we have to resort to a database-specific function. Aaargggghh!

Typical use of this function:

```
...
$sql = 'UPDATE {'$DB->prefix}table ' .
      'SET message = ' . $DB->concat('message', "' " . $DB->escape("addition to
message\n")."' ").' ' .
      'WHERE table_id = ' . $some_id;
```

```
$DB->exec($sql);
```

```
...
```

and that's exactly the kind of hairy code I'd like to stay away from, because of the necessary delicate balancing of quotes and the amount of thought it requires to get a query right. Obviously it is much easier (and less quote-error prone and almost elegant) to do something like this:

```
$record = db_select('tablename','message',array('table_id' => $some_id));
$value = $record['message'] . "addition to message\n";
db_update('tablename',array('message' => $value),array('table_id' => $some_id));
```

Note that 'Standard-SQL' would yield almost as much trouble with concatenation, even if it were possible to use in MySQL, e.g.:

```
...
$sql = 'UPDATE {'$DB->prefix}table ' .
      "SET message = message || '" . $DB->escape("addition to message\n")."'
      'WHERE table_id = ' . $some_id;
$DB->exec($sql);
...
```

No matter what: this is ugly. The reason I still want to use this kind of code is that (much) more expensive to use series of SELECT / UPDATE statements with concatenation in PHP, not to mention the fact that using two separate SQL-statements introduces race conditions, so there. Alas this is database specific.

Note that some of the quote-hell can be dealt with via [db_escape_and_quote\(\)](#):

```
$sql = 'UPDATE {'$DB->prefix}table ' .
      'SET message = ' . $DB->concat('message',db_escape_and_quote("addition to
message\n")).' ' .
      'WHERE table_id = ' . $some_id;
$DB->exec($sql);
```

- **TODO** perhaps extend this function to accept more than 2 strings?

bool function DatabaseMysqli::connect(\$db_server, \$db_username, \$db_password, \$db_name) [line 144]

Function Parameters:

- *string* **\$db_server** database server, e.g. 'localhost' or 'db.example.com:3306'
- *string* **\$db_username** part of credentials
- *string* **\$db_password** part of credentials
- *string* **\$db_name** database to use, e.g. 'was'

connect to the database server and open the database

this opens a connection to the database, perhaps sets some connection parameters and subsequently selects the requested database. If anything goes wrong, FALSE is returned and additional information can be retrieved from the variables \$this->errno and \$this->error.

Note that for 4.1.x <= MySQL < 5.5.3 we use charset utf8 for the connection, whereas for

5.5.3 and up we use the utf8mb4 charset which can handle UTF-8 with upto 4-byte sequences.

- **TODO** weigh pros and cons of persistent database connections, perhaps add as config option?

int|bool function DatabaseMysqli::create_table(\$tabledef) [*line 464*]

Function Parameters:

- *array* **\$tabledef** a generic table definition (not database-specific)

create a table via a generic (non-MySQL-specific) table definition

this executes a MySQL-specific CREATE TABLE statement based on a generic table definition. The actual work is done in create_table_sql(), which makes it possible to see the result of converting a generic definition to an actual table; very handy while debugging.

- **TODO** document correct link for documentation of generic table definition 'tabledefs.php'

string|bool function DatabaseMysqli::create_table_sql(\$tabledef) [*line 491*]

Function Parameters:

- *array* **\$tabledef** a generic table definition (not database-specific)

create the MySQL-specific SQL statement to create a table via a generic table definition

this creates a MySQL-specific CREATE TABLE statement from a generic table definition. See [tabledefs.php](#) for more information about the format of this generic table definition.

Note that this routine takes the level of UTF-8-support into account; for level 3 we use charset 'utf8' and for level 4 'utf8mb4' (see also [connect\(\)](#)).

The foreign key constraint is now capable of naming the constraints with a unique (database wide) symbol. (Cures InnoDB-error 1005 (HY000) errno: 121). This symbol is the prefixed

tablename followed by either the specified key name or a 1-based integer uniquemaker per table.

- **TODO** document correct link for documentation of generic table definition 'tabledefs.php'
- **TODO** find a way to deal with the enum values: where do we keep them? Or do we keep them at all?

int/bool function DatabaseMysqli::drop_table(\$tablename) [line 428]

Function Parameters:

- *string* **\$tablename** the name of the table to drop (prefix will be added automatically)

unconditionally drop the specified table

bool function DatabaseMysqli::dump(&\$data, [\$drop = TRUE], [\$tables = '']) [line 745]

Function Parameters:

- *string* **&\$data** receives the dump of the tables
- *bool* **\$drop** if TRUE add code to drop the table before the data definition
- *mixed* **\$tables** array with names of tables to dump, empty (string, array) means all our tables

make a text dump of our tables in the database suitable for backup purposes

This creates a text dump of the selected tables in parameter \$data. If \$tables is empty we dump all the tables in the database that start with 'our' prefix (there could be other websites using the same table with another prefix, we won't dump those). If \$tables is an array, it is assumed to be an array with table names without our prefix. In this case we will prepend the prefix. If parameter \$drop is TRUE, we add code to drop the table from the database (if it exists) before we recreate it.

If there were no errors, we return TRUE (and \$data contains the dump). If errors were encountered, we return FALSE and \$this->errno and \$this->error can tell the caller what happened. If there were errors, \$data is undefined.

Strategy is as follows. First we make a valid list of tables to dump, either by asking the database for a list of tables LIKE "{\$prefix}%" or by manipulating and validating

the array \$tables that was provided by the caller.

Subsequently we let the database generate a CREATE TABLE statement and then we step through the data (if any) and add it to \$data.

Note MySQL is quite liberal in what it accepts as field values. However, I try to generate INSERT INTO-statements as clean as possible by NOT quoting numeric values. HTH. It still is a MySQL-specific dump, though. You cannot simply use the result 'as-is' to migrate to another database.

string|bool function DatabaseMysqli::escape(\$unesaped_string) [line 223]

Function Parameters:

- *string* **\$unesaped_string** the string to escape

escape special characters in string

this makes sure that dangerous characters like single quotes are properly escaped. this routine has a special twist because of the limited support for UTF-8 in some MySQL-versions. If support is limited (\$this->utf_support equals 3), we strip any 4-byte UTF-8 characters from the string and we replace those with the generic substitution character U+FFFD (see als [mysqli_utf8mb3\(\)](#)).

The (bold) assumptions here are:

1. the \$unesaped_string is in fact proper UTF-8 (see [utf8_validate\(\)](#)), and
2. all strings that are headed for the database are funneled through this routine, always.

int|bool function DatabaseMysqli::exec(\$sql) [line 317]

Function Parameters:

- *string* **\$sql** valid SQL statement

execute an action query and return the number of affected rows

this method should be used to execute SQL-statements that do NOT return a result set, use query() for that. This method works well for INSERTs, DELETEs and UPDATEs. If there is an error this method returns FALSE. Otherwise it returns the number of affected lines. Note that there is a difference between 0 affected lines and FALSE.

int|bool function DatabaseMysqli::last_insert_id([\$table_name = "], [\$field_name = "]) [*line 365*]

Function Parameters:

- *string* **\$table_name** (optional) tablename (unused in MySQL)
- *string* **\$field_name** (optional) fieldname (unused in MySQL)

retrieve the most recent automatically inserted id ('auto_increment')

this method returns the id that was automatically generated in the previous INSERT-query or 0 if the previous query was not an INSERT query.

Note: as per the MySQL manual this function returns an int and not a bigint. If the auto_increment field is a bigint, this method returns an incorrect result. There is a work-around (e.g. SELECT LAST_INSERT_ID()) but this is not the way it works in Website@School; all id's are simple int's and not bigint's, so there is no need to generate yet another query after every insert.

Note that this method can be called with a table name and a field name. This is a hook for future expansion; this MySQL-driver does not actually use it. However, it is handy to always call this method with table and field name to make adding a new database driver easier.

string function DatabaseMysqli::mysqli_utf8mb3(\$utf8str) [*line 924*]

Function Parameters:

- *string* **\$utf8str** valid UTF-8 encoded string

message string to contain only 3-byte UTF8-sequences

this replaces an otherwise perfectly valid 4-byte UTF-8 sequence in \$utf8str with a 3-byte UTF-8 sequence equivalent with the Unicode replacement character U+FFFD.

The effect is that it leaves a hint that there used to be some character instead of silently discarding 4-byte sequences which MySQL does.

int|bool function DatabaseMysqli::mysqli_utf8_support(\$db_link) [*line 894*]

Function Parameters:

- *resource* **\$db_link** the MySQL connection

determine the level of UTF-8 support based on MySQL-server version

MySQL support for UTF-8 was non-existent before 4.1.x and limited until 5.5.3. In this context 'limited' means: only the Basic Multilingual Plane (U+0000 ... U+FFFF) is supported, i.e. a maximum of 3-byte sequences per character.

As of 5.5.3 the full UTF-8 specification according to RFC 3629 is implemented. MySQL now has 'invented' yet another proprietary name for this character set: 'utf8mb4' (WTF?), and introduces the alias 'utf8mb3' for the pre 5.5.3 limited support for 'utf8' (WTF??), hinting that the meaning of 'utf8' may change in future versions to indicate 'utf8mb4' (WTF???). IM(NS)HO this is yet another reason to go looking for a decent replacement for MySQL. YMMV.

This routine returns exactly one of the values below (based on the server version).

- 0: there is no UTF-8 support available in this server
- 3: the limited 3-byte sequences are supported
- 4: full support for 4-byte sequences available, but using the stupid ad-hoc name 'utf8mb4' or the value FALSE if version information could not be obtained.

As a side effect, we record the server version information in `$this->db_version` (which is handy when creating a backup and also for debugging purposes).

object|bool function DatabaseMysqli::query(\$sql, [\$limit = "], [\$offset = "]) [*line 396*]

Function Parameters:

- *string* **\$sql** valid SQL statement
- *int* **\$limit** optional limitation of the number of records returned
- *int* **\$offset** optional number of records to skip; \$offset = 0 means start with the first

execute a select query and return a result set

this method should be used to execute SQL-statements that do return a result set: SELECT, SHOW, EXPLAIN and DESCRIBE. Use `exec()` for action queries. If there is an error this method returns FALSE. Otherwise it returns a result set in the form of a DatabaseMysqliResult object.

if parameters `$limit` and `$offset` are set the result set contains at most `$limit` rows, starting at offset `$offset`. it is OK to specify just `$limit`; `$offset` is taken into account only when `$limit` is specified too. Note that different databases have a different syntax for limiting the number of returned records. MySQL supports both 'LIMIT limit OFFSET offset' (which we use here) and 'LIMIT offset,limit'.

the LIMIT-clause is blindly appended to the SQL-statement; it is up to the caller to decide whether specifying a limit and an offset make sense or not.

bool function DatabaseMysqli::table_exists(\$tablename) [*line 439*]

Function Parameters:

- *string* **\$tablename** name of the table to check (prefix will be added automatically)

see if the named table exists

Class DatabaseMysqliResult

[*line 1056*]

MySQL database result

This implements access to database result sets

- **Package** wascore

DatabaseMysqliResult::\$db_link

resource = [*line 1070*]

- **Var** \$db_link is the database link identifier

DatabaseMysqliResult::\$errno

integer = [*line 1061*]

- **Var** the error number generated by the latest mysqli command

DatabaseMysqliResult::\$error

string = [line 1064]

- **Var** the error message generated by the latest mysqli command

DatabaseMysqliResult::\$num_rows

integer = [line 1067]

- **Var** the number of rows in the result set

DatabaseMysqliResult::\$result

resource = [line 1058]

- **Var** the resource associated with the result set

Constructor *void* function DatabaseMysqliResult::DatabaseMysqliResult(\$db_link, \$result) [line 1076]

Function Parameters:

- *resource* **\$db_link** the resource associated with the result set
- **\$result**

constructor

bool function DatabaseMysqliResult::close() [line 1089]

free the memory associated with the result set

array function DatabaseMysqliResult::fetch_all() [line 1121]

fetch all rows as a 0-based array of 0-based enumerated arrays

array function DatabaseMysqliResult::fetch_all_assoc([\$keyfield = ""]) [*line 1141*]

Function Parameters:

- *string* **\$keyfield** field to use as the key in the returned array or empty for 0-based numeric array key

fetch all rows as an array (0-based or keyed) of associative arrays

This returns an array of assoc arrays (one per record). If \$key is not empty, we use the contents of the field as the array key, otherwise we simply use a 0-based numeric key. By specifying a single unique field (e.g. the primary key) all records in the array can be accessed via their unique value.

array/bool function DatabaseMysqliResult::fetch_row() [*line 1101*]

fetch the next result row as a 0-based enumerated array

array/bool function DatabaseMysqliResult::fetch_row_assoc() [*line 1111*]

fetch the next result row as a associative array

Class DatabaseMysqliResult

[*line 1036*]

MySQL database result

This implements access to database result sets

- **Package** wascore

DatabaseMysqliResult::\$errno

integer = [*line 1041*]

- **Var** the error number generated by the latest mysql command

DatabaseMysqlResult::\$error

string = [line 1044]

- **Var** the error message generated by the latest mysql command

DatabaseMysqlResult::\$num_rows

integer = [line 1047]

- **Var** the number of rows in the result set

DatabaseMysqlResult::\$result

resource = [line 1038]

- **Var** the resource associated with the result set

Constructor *void* function DatabaseMysqlResult::DatabaseMysqlResult(\$result) *[line 1053]*

Function Parameters:

- *resource* **\$result** the resource associated with the result set

constructor

bool function DatabaseMysqlResult::close() *[line 1065]*

free the memory associated with the result set

array function DatabaseMysqlResult::fetch_all() [*line 1095*]

fetch all rows as a 0-based array of 0-based enumerated arrays

array function DatabaseMysqlResult::fetch_all_assoc([\$keyfield = ""]) [*line 1115*]

Function Parameters:

- *string* **\$keyfield** field to use as the key in the returned array or empty for 0-based numeric array key

fetch all rows as an array (0-based or keyed) of associative arrays

This returns an array of assoc arrays (one per record). If \$key is not empty, we use the contents of the field as the array key, otherwise we simply use a 0-based numeric key. By specifying a single unique field (e.g. the primary key) all records in the array can be accessed via their unique value.

array/bool function DatabaseMysqlResult::fetch_row() [*line 1077*]

fetch the next result row as a 0-based enumerated array

array/bool function DatabaseMysqlResult::fetch_row_assoc() [*line 1086*]

fetch the next result row as a associative array

Class Email

[*line 45*]

Email implements a simple interface to send mail

This class can be used to send mail from Website@School, e.g. alerts, new passwords, feedback to the project (with attached translations), etc.

```
Typical use: require_once('email.class.php');
$mailer = new Email;
$mailer->set_mailto($email,$name);
$mailer->set_subject($subject);
$mailer->set_message($message);
$mailer->add_attachment($data,$name);
$mailer->send();
```

- **Package** wascore

Email::\$attachments

array = array() [line 68]

- **Var** \$attachments array of arrays with attachment properties: body, name, mimetype, etc.

Email::\$charset

string = UTF-8 [line 77]

- **Var** \$charset default character set to use in display names and subject

Email::\$eol

string = \r\n [line 74]

- **Var** \$eol end of line character(s), usually CR + LF

Email::\$headers

array = array() [line 62]

- **Var** \$headers associative array with field names and field values of additional headers

Email::\$mailcc

array = array() [line 56]

- **Var \$mailcc** contains an array of arrays containing addr+name for Cc: (array of addresses)

Email::\$mailfrom

array = array() [line 47]

- **Var \$mailfrom** contains addr and name for From: (single address)

Email::\$mailreplyto

array = array() [line 50]

- **Var \$mailreplyto** contains addr and name for Reply-To: (single address)

Email::\$mailto

array = array() [line 53]

- **Var \$mailto** contains addr and name for To: (single address)

Email::\$max_length

int = 76 [line 83]

- **Var \$max_length** limit for line length

Email::\$messages

array = array() [line 65]

- **Var** \$messages array of arrays with message properties: body, mimetype, charset, encoding

Email::\$minimal

bool = FALSE [line 80]

- **Var** \$minimal default flag limiting the literal representation in [rfc2047_qchar\(\)](#)

Email::\$related

bool = FALSE [line 71]

- **Var** \$related TRUE indicates multipart/related mail rather than multipart/mixed

Email::\$subject

string = [line 59]

- **Var** \$subject contains the message subject

Constructor *void* function Email::Email() [line 89]

constructor resets all variables to a known (default) state

void function Email::add_attachment(\$attachment, \$name, [\$mimetype = 'application/octet-stream'], [\$charset = 'UTF-8'], [\$encoding = 'base64'], [\$description = ''], [\$disposition = 'attachment']) [line 265]

Function Parameters:

- *string* **\$attachment** presumably 8bit data to attach to the message

- *string* **\$name** the suggested filename to use when receiving the attachment
- *string* **\$mimetype** type of the content, usually the generic 'application/octet-stream'
- *string* **\$charset** character set to use (only applicable when \$mimetype indicates 'text')
- *string* **\$encoding** the desired encoding (defaults to base64 because we expect binary data)
- *string* **\$description** optional description of the attachment
- *string* **\$disposition** either 'inline' or 'attachment'

add an attachment

This simply adds an attachment with associated properties. Multiple attachments can be added by calling this routine multiple times.

The attachments added via this routine are simply added to the current list of attachments in `$this->attachments`. There is also another routine (`@link add_related()`) which takes care of attachments in a multipart/related context. Both routines do add to the same array.

void function Email::add_mailcc(\$addr, [\$name = "]) [line 186]

Function Parameters:

- *string* **\$addr** the address eg. 'acackl@example.com'
- *string* **\$name** the name, eg. 'Amelia Cackle'

add an address and name for the Cc: header

Embedded CRs LFs "<" and ">" are removed from \$addr, and the result is stored, together with \$name. Note that this function can be called multiple times, where each call adds an address to the list.

void function Email::add_message(\$message, [\$mimetype = 'text/plain'], [\$charset = 'UTF-8'], [\$encoding = 'quoted-printable']) [line 239]

Function Parameters:

- *string* **\$message** content to send
- *string* **\$mimetype** type of the content, usually 'text/plain'
- *string* **\$charset** character set to use

- *string* **\$encoding** the desired encoding (defaults to quoted-printable because we expect text)

add an (alternative version of) message

This simply stores an alternative message body until it can be combined (to a multipart/alternative) and subsequently sent.

Note that according to RFC1341 the sender should place body parts in increasing order of preference, ie. first text/plain, then text/html and finally application/pdf. However, it is up to the caller to call and [add_message\(\)](#) in the correct order.

string function Email::add_related(\$attachment, \$name, [\$mimetype = 'application/octet-stream'], [\$charset = 'UTF-8'], [\$encoding = 'base64'], [\$description = ''], [\$disposition = 'inline']) [*line 310*]

Function Parameters:

- *string* **\$attachment** presumably 8bit data that relates to the message
- *string* **\$name** the suggested filename to use when receiving the attachment
- *string* **\$mimetype** type of the content, usually the generic 'application/octet-stream'
- *string* **\$charset** character set to use (only applicable when \$mimetype indicates 'text')
- *string* **\$encoding** the desired encoding (defaults to base64 because we expect binary data)
- *string* **\$description** optional description of the attachment
- *string* **\$disposition** either 'inline' or 'attachment'

add a related attachment

This adds a related attachment with associated properties. Multiple related attachments can be added by calling this routine multiple times.

The attachments added via this routine are added to the current list of attachments in `$this->attachments`, but they are handled in a slightly different way once the mail body is constructed. There is another routine (`@link add_attachment()`) which takes care of 'plain' attachments (which usually leads to a multipart/mixed message). Both routines add to the same array.

Note 1: all related attachments are assigned a (hopefully globally) unique Content-ID. This ID can be used to refer to this message part, e.g. via HTML img-tags `src="cid:$content_id"`. See also RFC2111. The Content-ID adheres to the same rules as a Message-ID, hence we use the same routine for both.

Note 2: once a related attachment is added via this routine, the flag `$this->related` is set to

TRUE, making sure we eventually generate a multipart/related rather than a multipart/mixed message.

- Uses [Email::rfc5322_message_id\(\)](#)

string function Email::boundary() [*line 1067*]

construct a unique boundary for use within this mail message

the combination of our pid, the current date up to a second and a unique number within this run should provide enough uniqueness within this mail message. The maximum length of the resulting string is at most 44 characters (assuming signed integers take at most 11 positions in decimal form). However, as a rule `get_unique_number()` starts with a single '1' so a practical max length of 34 is far more likely.

bool function Email::encode_message(&\$body, &\$headers, [\$stoplevel = FALSE]) [*line 458*]

Function Parameters:

- *string* **&\$body** receives the email body
- *array* **&\$headers** receives necessary header fields (if `$stoplevel == TRUE`)
- *bool* **\$stoplevel** determines where the headers are stored: in `$body` or `$headers`

encode the main message, optionally including 1 or more alternative versions

this encodes the main message as a single body part. If there are alternative versions of the message, all variants are wrapped in a multipart/alternative body part.

Depending on the `$stoplevel` flag the associated headers are either appended to the `$body` (`$stoplevel==FALSE`) OR added to the `$headers` array (`$stoplevel==TRUE`).

The actual encoding is done in [encode_part\(\)](#).

- Uses [Email::encode_part\(\)](#)

bool function Email::encode_part(&\$source, &\$body, &\$headers, [\$stoplevel = FALSE]) *[line 500]*

Function Parameters:

- *array* **&\$source** holds either a message or an attachment + associated properties
- *string* **&\$body** receives the email body
- *array* **&\$headers** receives necessary header fields (if \$stoplevel == TRUE)
- *bool* **\$stoplevel** determines where the headers are stored: in \$body or \$headers

encode an email body part with headers and all

this encodes the message or attachment in \$source with the correct mime type and other headers. Depending on the \$stoplevel flag, the headers end up in the \$body (as part of a multipart mime message) OR in the \$headers array (if the caller has decided that there is only a single body part in this email message).

The necessary headers are computed in a separate routine [encode_part_headers\(\)](#).

- **Usedby** [Email::encode_message\(\)](#)
- **Uses** [Email::encode_part_headers\(\)](#)

array function Email::encode_part_headers(&\$source) *[line 550]*

Function Parameters:

- *array* **&\$source** points to a message or an attachment

construct necessary headers for a MIME body part

this routine constructs the following headers:

- Content-Type
- Content-Transfer-Encoding
- Content-Description (optional)
- Content-Disposition (optional)
- Content-ID (optional)

based on the information in \$source. Basically this is copying and combining the information of the elements in \$source. However, there is 1 exception: if \$source appears to

be a 7-bit ASCII message with lines shorter than 78 characters, the charset in Content-Type type is overruled and set to US-ASCII and the Content-Transfer-Encoding is set to 7bit, all in an attempt to keep simple messages as readable as possible.

- Used by [Email::encode_part\(\)](#)

bool function Email::is_7bit(&\$source) [line 871]

Function Parameters:

- *string &\$source* the text to examine

a small utility routine to determine if a string has only 7bit characters

- Used by [Email::rfc2047_qstring\(\)](#)
- Used by [Email::send\(\)](#)

bool function Email::prepare_body(&\$body, &\$headers) [line 399]

Function Parameters:

- *string &\$body* receives the complete mail body
- *array &\$headers* receives the necessary mail header fields

construct the full mail body and the necessary top-level mail header fields

this routine examines the messages that were added to Email (in `$this->messages`): that array could contain 0, 1 or more versions of the message to send. Also this routine looks at the 0, 1 or more plain or related attachments that were added to `$this->attachments`.

If there are multiple entries in `$this->messages`, those are wrapped in a multipart/alternative body part. If there is only a single message it becomes one of the body parts in its own right. If there are any attachments, both the message (which might be multipart/alternative) and the attachments are combined using either multipart/mixed (there

are only plain attachments) or multipart/related (there was at least one related attachment).

The resulting mail body is returned to the caller via `&$body`. Hopefully this saves some memory moving that possibly big string around. The headers that need to go in the mail headers are returned via `&$headers`. It is the responsibility of the caller to make sure that these headers end up in the correct place.

`void function Email::reset_all()` [line 97]

reset all variables to their default values

`string function Email::rfc2047_qchar($c, &$required_len, [$minimal = FALSE])` [line 841]

Function Parameters:

- `int $c` the character to encode
- `int &$required_len` returns the space required for this encoded char (UTF8-aware)
- `bool $minimal` if TRUE, only [0-9A-Za-z] use literal representation, otherwise encoding is more relaxed

encode an 8-bit byte according to Q-encoding in RFC2047

This routine encodes a single integer ASCII code into either

- literal representation
- generic 8bit representation, ie. "=" followed by 2 (uppercase) hexdigits
- an underscore character

If `$minimal` is FALSE, all printable ASCII characters from 33 "!" to 126 "~" except 61 "=", 63 "?" and 95 "_" use literal representation. Character 32 " " is represented as an underscore (for improved readability/deciphering).

If `$minimal` is TRUE, only digits "0" - "9" and letters "A" - "Z" and "a" - "z" use literal representation and character 32 " " uses generic 8bit encoding "=20".

The latter case yields only digits, letters, equal-sign and question mark, which should travel undisturbed through any mail transfer agent.

There is a special situation when encoding UTF8 where characters can span multiple octets. The length of such a sequence can be determined by the number of most significant 1's in a row in the first octet. If `$c` is the first octet of a UTF8-sequence, we tell the caller the total length of the encoded sequence, not just the length of the encoded 1st octet (which would always be 3). This forces the caller to start a new 'encoded-word' with enough room for the complete sequence if necessary, preventing a multi-octet sequence to span two 'encoded-words'. Note that characters in a UTF8-tail yield length 3, even when more UTF8-tail octets

follow. That is OK because the first character already 'reserved' the space when the first octet was processed.

Here is a small truth table for sequence lengths (see also RFC3629). bit pattern range len comments
0xxx.xxxx 0-127 3 ASCII 10xx.xxxx 128-191 3 octet is part of UTF8-tail
110x.xxxx 192-223 6 UTF8-2, beginning of a sequence of 2 octets 1110.xxxx 224-239 9 UTF8-3, beginning of a sequence of 3 octets 1111.0xxx 240-247 12 UTF8-4, beginning of a sequence of 4 octets 1111.10xx 248-251 3 sequence of 5 characters not defined in RFC3929, settle for length 3 1111.110x 252-253 3 sequence of 6 characters not defined in RFC3929, settle for length 3 1111.111x 254-255 3 no sequence at all, settle for length 3

Note that if \$c is NOT UTF8 but say ISO-5988-1, the worst that can happen is that a perfectly valid single octet character in the range 192-247 would indicate a length of more than the necessary 3, pushing up to 4 characters to the next 'encoded-word'. Oh well, I can live with that.

References: <http://www.ietf.org/rfc/rfc2047.txt>, <http://www.ietf.org/rfc/rfc3629.txt>.

- Used by [Email::rfc2047_qstring\(\)](mailto:rfc2047_qstring())

string function Email::rfc2047_qstring(\$source, &\$remaining, [\$charset = "UTF-8"], [\$minimal = FALSE], [\$max_length = 76], [\$eol = "\r\n"]) [*line 730*]

Function Parameters:

- *string* **\$source** the string to encode (could be 7bit)
- *int* &**\$remaining** the number of bytes remaining on the current output line (not counting any CR+LF)
- *string* **\$charset** indicates character set used in \$source
- *bool* **\$minimal** if TRUE, rfc2047_qchar() limits literal encoding to digits and letters
- *int* **\$max_length** limit on output line length (and indirect of the 'encoded-word' length) to max 76 (75)
- *string* **\$eol** the end of line character(s), default as per RFC5322 (RFC822) is CR chr(13) + LF chr(10)

encode a string according to RFC2047 (Message Header Extensions for Non-ASCII Text)

This routine encodes \$source according to RFC2047 (Message Header Extensions for Non-ASCII Text) using the 'Q'-encoding (somewhat comparable to quoted_printable). However, if the \$source uses only harmless 7bit characters and falls within the limit of \$remaining characters it is returned unchanged and the number of \$remaining characters is updated accordingly.

In all other cases (\$source contains bytes > 127, \$source is longer than \$remaining, etc.) this encodes the string into 'encoded-word's of max 75 chars. These 'encoded-word's look like this: "=?" charset "?" encoding "?" encoded-text "=?" with 'encoding' always equal to "Q" (similar to quoted printable). The actual encoding of characters is done in [rfc2047_qchar\(\)](#). The boolean flag \$minimal can be used to limit the literal representation to only digits and letters, using generic 8bit encoding by setting it to TRUE.

If multiple 'encoded-word's are necessary, they are separated from each other by a folding space, i.e. newline (using \$eol) followed by a normal space (ASCII 32). The end result never ends with such a folding space; the returned value always ends with the "=?" of the last 'encoded-word'.

Note 1: I found it quite hard to read the combination of RFC5322 and RFC2047 because I had some trouble distinguishing the rules for RFC5322-type headers. I finally settled for this simplified set

- From:, To:, Cc: and Reply-To: are all of type 'mailbox' to me (KISS, no 'group's and stuff)
- A 'mailbox' can be either written as ['display-name'] "<" addr-spec ">" OR as
addr-spec "(" ctext ")", which both allow for FWS (folding white space)
- A Subject: field is simply 'unstructured', which also allows for FWS

Furthermore, this routine simply encodes non-ASCII text and therefore makes no assumptions about the contents of \$source; it is the caller's responsibility to make sure that the 'ctext' or 'unstructured' or 'display-name' conforms to RFC5322.

Note 2: I have not implemented fancy and streamlined code to minimise the amount of encoded characters, ie. leaving pure ASCII-words unencoded and encoding only non-ASCII words or words containing "=?" because I could not invest that amount of time.

Maybe in a later version... (famous last words). I did optimise for short and pure and simple ASCII strings because I expect that generated Subject: headers and other headers will be ASCII most of the time. We'll see how that works out.

Note 3: Quirk: if initially there is not enough space for the shortest possible 'encoded-word', we insert a FWS even if \$source has no WSP at that point. Basically it means that we add a character to the result. This may or may not be a problem for the caller. OTOH: the caller should provide enough space in the first place, so there.

References: see <http://www.ietf.org/rfc/rfc2047.txt> and <http://www.ietf.org/rfc/rfc5322.txt>.

- **TODO** maybe optimise this routine to let pure ASCII-words through unencoded (in a later version)
- **Usedby** [Email::rfc5322_address\(\)](#)
- **Usedby** [Email::send\(\)](#)
- **Uses** [Email::rfc2047_qchar\(\)](#)
- **Uses** [Email::is_7bit\(\)](#)

string function Email::rfc5322_address(\$addr_spec, &\$remaining, [\$display_name = "], [\$legacy = FALSE], [\$charset = "UTF-8"], [\$minimal = FALSE], [\$max_length = 76], [\$eol = "\r\n"]) [*line 933*]

Function Parameters:

- *string* **\$addr_spec** is an address of the form 'local-part' "@" 'domain' (RFC5322 section 3.4.1)
- *int* **&\$remaining** indicates how much space is left on the current output line
- *string* **\$display_name** is the human readable name associated with \$addr_spec
- *bool* **\$legacy** if TRUE, output is in the legacy format 'addr-spec' "(" 'ctext' ")"
- *string* **\$charset** indicates character set used in \$display_name
- *bool* **\$minimal** if TRUE, eventually rfc2047_qchar() limits literal encoding to digits and letters
- *int* **\$max_length** limit on output line length (and indirect of the 'encoded-word' length) to max 76 (75)
- *string* **\$eol** the end of line character(s), default as per RFC5322 (RFC822) is CR chr(13) + LF chr(10)

construct an address field according to RFC5322 (RFC822)

This routine constructs an address according to (simplified) rules in RFC5322 section 3.4. Depending on the \$legacy flag, the parameters are used to construct either an 'angle-addr' DQUOTE \$display_name DQUOTE SPACE "<" \$addr_spec ">" or an address with the display-name "hidden" in a comment \$addr_spec SPACE "(" \$display_name ")"

Basically the \$addr_spec is not modified; it is assumed that this string obeys the rules for 'addr-spec' in RFC5322. Specifically we do not encode this information (with [rfc2047_qstring\(\)](#) or otherwise). However, we DO strip any CR and/or LF characters because these might cause problems lateron (eg. an unwanted extra blank line in the mail headers). Also, we

definitely do not want to have angle brackets, so we remove those too, just to be sure.

The `$display_name` can be modified. Reading RFC5322 yields the following (simplified) rules. 'display-name' => 'phrase' => 1*'word'; 'word' => 'atom' | 'quoted-string'. In other words: it is allowed to use a quoted string, ie.

- a DQUOTE, followed by
- a string with printable ASCII characters not being DQUOTE or the quote character backslash, followed by
- a DQUOTE.

Note that FWS (folding white space) is allowed between the two DQUOTES. This leads an easy way out of to stripping DQUOTES and backslashes before the 'display-name' is encoded and/or folded.

If the legacy-flag is set, we use the parameter `$display_name` to construct a (simplified) comment, ie.

- a "(", followed by
- a string of ctext characters not containing "(", ")" or a backslash, followed by
- a ")".

Here, too, folding whitespace is allowed between the opening "(" and closing ")". This variant leads the easy way out of stripping parentheses and backslashes before the 'display-name' is encoded and/or folded as a comment.

Note: All this cleaning up of CR, LF, DQUOTE, etc. does NOT weed out other CTLs (ASCII 0...ASCII 31).

- **Usedby** [Email::send\(\)](#)
- **Uses** [Email::rfc2047_qstring\(\)](#)

string function `Email::rfc5322_message_id()` [*line 1023*]

construct a message-id conforming to RFC5322 (RFC2822, RFC822)

This constructs a message id according to specifications in section 3.6.4 in RFC5322 Internet Message Format (October 2008), see <http://www.ietf.org/rfc/rfc5322.txt>.

Note that the 'id-left' in the 'msg-id' also contains the remote IP-address. This could be an IPv4 address in the usual dotted-decimal form but it could also be an IPv6 address like '::1' (3 characters) or '[FEDC:BA98:7654:3210:FEDC:BA98:7654:3210]' (41 characters). The total maximum-length of 'id-left' may add up to 11 (32-bit signed pid) + 1 (dot) + 41 (full IPv6 w/ brackets) + 1 (hash) + 11 (32-bit signed remote port) + 1 (dot) + 14 (date/time) + 11 (signed unique number) = 91 characters. This is longer than the recommended linelength of 78 characters, but the absolute maximum of 998 characters will probably NOT be reached. OTOH: we do not actually check for huge domain names and/or server names. Oh well.

Note that we massage the IPv6 address by replacing any ':', '[' and ']' with '!', '{' and '}' respectively because the former are not allowed in a dot-atom-text. As a matter of fact we translate most 'specials' to 'atext' (RFC5322 3.2.3). Notable exception: the dot stays.

- **TODO** how about UTF-8 hostnames? Mmmm...
- **Usedby** [Email::send\(\)](#)
- **Usedby** [Email::add_related\(\)](#)

bool function Email::send() [line 366]

send the message using the prepared information (To:, Subject:, the message and attachments etc.)

this routine prepares the raw message + necessary headers and subsequently sends the message.

- **Uses** [Email::rfc5322_message_id\(\)](#)
- **Uses** [Email::rfc5322_address\(\)](#)
- **Uses** [Email::rfc2047_qstring\(\)](#)
- **Uses** [Email::is_7bit\(\)](#)

bool function Email::send_body(&\$message, &\$message_headers) [line 605]

Function Parameters:

- *string* **&\$message** contains the completely prepared message (including multipart etc.)
- *array* **&\$message_headers** contains headers that need to be added the message's headers

actually send a prepared mail body (with headers) to recipient(s)

this routine constructs the necessary header fields ('To', 'Subject', 'From', 'Reply-To', 'Cc', 'X-Mailer', etc. and subsequently sends \$message to the recipient(s).

Note that there are several sources for headers: we start with `$this->headers`, followed by the list of headers mentioned above and ending with the headers in `$message_headers`. This means that `$message_headers` has the last word in headers; existing headers in `$headers` will be overwritten with headers present in `$message_headers`. As a rule at most the following headers are set in `$message_headers`: 'MIME-Version', 'Content-Type', 'Content-Transfer-Encoding' and 'Message-ID':.

`void function Email::set_header($name, [$value = ""]) [line 352]`

Function Parameters:

- *string* **\$name** is the name of the header field
- *string* **\$value** is the contents of the header field

manually add a header to the mail message

This adds a header to the message headers. Possible candidates are

- 'Priority' with possible values (from RFC2156): "normal" | "non-urgent" | "urgent"
- 'Importance' with possible values (from RFC2156): "low" | "normal" | "high"

Note that the following headers may be overwritten in the course of constructing the message to send (see [send\(\)](#)):

- To: (depending on how `mail()` handles the first parameter internally)
- Subject: (depending on how `mail()` handles the second parameter internally)
- From:
- Reply-To:
- Cc:
- X-Mailer:
- Message-ID:
- MIME-Version:
- Content-Type:
- Content-Transfer-Encoding:

It is possible to use tricks such as using a different capitalisation to defeat this. (I said it was a simple class, didn't I?)

- **TODO** should we bring the Capi-Tali-Sation of `$name` in line with the default capitalisation in the list above?

`void function Email::set_mailfrom($addr, [$name = ""]) [line 136]`

Function Parameters:

- *string* **\$addr** the address eg. 'webmaster@example.com'
- *string* **\$name** the name, eg. 'Exemplum Primary School'

record the address and the name for the From: header

Embedded CRs LFs "<" and ">" are removed from \$addr, and the result is stored, together with \$name.

void function Email::set_mailreplyto(\$addr, [\$name = "]) [line 152]

Function Parameters:

- *string* **\$addr** the address eg. 'info@example.com'
- *string* **\$name** the name, eg. 'Exemplum Primary School'

record the address and the name for the Reply-To: header

Embedded CRs LFs "<" and ">" are removed from \$addr, and the result is stored, together with \$name.

void function Email::set_mailto(\$addr, [\$name = "]) [line 168]

Function Parameters:

- *string* **\$addr** the address eg. 'hparkh@example.com'
- *string* **\$name** the name, eg. 'Helen Parkhurst'

record the address and the name for the To: header

Embedded CRs LFs "<" and ">" are removed from \$addr, and the result is stored, together with \$name.

void function Email::set_message(\$message, [\$mimetype = 'text/plain'], [\$charset = 'UTF-8'], [\$encoding = 'quoted-printable']) [line 214]

Function Parameters:

- *string* **\$message** content to send

- *string* **\$mimetype** type of the content, usually 'text/plain'
- *string* **\$charset** character set to use
- *string* **\$encoding** the desired encoding (defaults to quoted-printable because we expect text)

set the (primary) message

This simply stores the message primary body until it can be sent. Many times there is just a single message (perhaps with attachments), but it is possible to add an alternative message via [add_message\(\)](#). In that case the array `$this->messages[]` holds more than 1 variation of the message. This routine always stores 'the' message, i.e. if there was a set of messages before, those will be removed and an array with a single message remains.

Note that according to RFC1341 the sender should place body parts in increasing order of preference, ie. first text/plain, then text/html and finally application/pdf. However, it is up to the caller to call and [add_message\(\)](#) in the correct order.

void; function Email::set_subject(\$subject) [*line 122*]

Function Parameters:

- *string* **\$subject**

store the subject of the mail message

Embedded CRs and LFs in `$subject` are removed and the result is stored until message send time.

Class FileManager

[*line 67*]

File Manager

This class implements the File Manager.

This class is also used to browse files and images from FCKEditor. Distinction is made via the `$job` parameter in the constructor.

All the work is directed from the constructor, so it is enough to simply instantiate a new object

and let the constructor do the work. The only thing needed is an output object (see [AdminOutput](#)).

- **Package** wascore

FileManager::\$areas

null|array = NULL [line 75]

- **Var** \$areas holds all area records (for future reference) or NULL if not yet set

FileManager::\$current_directory

string = [line 84]

- **Var** \$current_directory links to the session-variable that holds the current working directory

FileManager::\$ext_allow_browse

bool|array = FALSE [line 93]

- **Var** \$ext_allow_browse holds browsable filename extensions (lowercase), FALSE (none) or TRUE (all)

FileManager::\$ext_allow_upload

bool|array = FALSE [line 90]

- **Var** \$ext_allow_upload holds uploadable filename extensions (lowercase), FALSE (none) or

TRUE (all)

FileManager::\$job

string = [line 72]

- **Var** \$job indicates how we are called (eg. as 'filemanager' or as 'filebrowser' or 'imagebrowser')

FileManager::\$output

object|null = NULL [line 69]

- **Var** \$output collects the html output

FileManager::\$show_thumbnails

bool = FALSE [line 96]

- **Var** \$show_thumbnails if TRUE we display files graphically (as a thumbnail), otherwise in table format

FileManager::\$sort

int = SORTBY_FILE_ASC [line 87]

- **Var** \$sort holds the current sort order in directory listings (default SORTBY_FILE_ASC)

FileManager::\$usergroups

null|array = NULL [line 78]

- **Var** \$usergroups holds all \$USER's group records (for future reference) or NULL if not yet set

FileManager::\$vpaths

array = array() [line 81]

- **Var** \$vpaths is a cache of virtual paths (see [vpath\(\)](#))

Constructor *void* function FileManager::FileManager(&\$output, [\$job = JOB_FILEMANAGER]) [line 112]

Function Parameters:

- *object* **&\$output** collects the html output
- *string* **\$job** indicates the mode: filemanager, filebrowser (FCKEditor) or imagebrowser (FCKEditor)

construct a FileManager object (called from /program/main_admin.php)

This initialises the FileManager, checks user permissions and finally dispatches the tasks. If the specified task is not recognised, the default task TASK_LIST_DIRECTORY is executed.

Note that many commands act on the directory contained in the SESSION-variable `current_directory`.

- **TODO** a nice filter for JOB_IMAGEBROWSER and also an alternative user interface for browsing/selecting images

bool/array function FileManager::allowed_extensions(\$allowed_extensions_list) [line 3062]

Function Parameters:

- *string* **\$allowed_extensions_list** comma-delimited string with allowable extensions (or empty string)

convert a comma-delimited list of allowable extensions to an array (or FALSE if none are allowed)

this converts the comma-delimited list of allowable filename extensions to an array with one element per allowable extension OR to a boolean with value FALSE if no allowable extensions are specified. The input is converted to lower case and also spaces and dots are removed (in anticipation of users entering the bare extension with the dot while we use the bare extension here. Also, it appears very natural to specify a list including spaces (which we don't want), so there.

int function FileManager::cmp_entries_bydate_asc(\$a, \$b) [line 2257]

Function Parameters:

- *array \$a* first entry
- *array \$b* second entry

callback for comparing two directory entries by mtime

Comparison between a file and a directory always shows directories first.

int function FileManager::cmp_entries_bydate_desc(\$a, \$b) [line 2283]

Function Parameters:

- *array \$a* first entry
- *array \$b* second entry

callback for comparing two directory entries by date (descending)

Comparison between a file and a directory always shows directories first, nevermind that we are sorting in descending order.

int function FileManager::cmp_entries_byfile_asc(\$a, \$b) [line 2165]

Function Parameters:

- *array \$a* first entry
- *array \$b* second entry

callback for comparing two directory entries by filename

Comparison between a file and a directory always shows directories first.

int function FileManager::cmp_entries_byfile_desc(\$a, \$b) [*line 2186*]

Function Parameters:

- *array* **\$a** first entry
- *array* **\$b** second entry

callback for comparing two directory entries by filename (descending)

Comparison between a file and a directory always shows directories first, nevermind that we are sorting in descending order.

int function FileManager::cmp_entries_bysize_asc(\$a, \$b) [*line 2206*]

Function Parameters:

- *array* **\$a** first entry
- *array* **\$b** second entry

callback for comparing two directory entries by size

Comparison between a file and a directory always shows directories first.

int function FileManager::cmp_entries_bysize_desc(\$a, \$b) [*line 2232*]

Function Parameters:

- *array* **\$a** first entry
- *array* **\$b** second entry

callback for comparing two directory entries by size (descending)

Comparison between a file and a directory always shows directories first, nevermind that we are sorting in descending order.

int function FileManager::cmp_groups(\$a, \$b) [*line 2122*]

Function Parameters:

- *array* **\$a** first array with groupdata (straight copy from database record)
- *array* **\$b** second array with groupdata (straight copy from database record)

callback for comparing two group records

This routine is used to order a list of groups by full_name, groupname.

bool function FileManager::delete_directory(\$path, \$directories, \$entries) [*line 1593*]

Function Parameters:

- *string* **\$path** the directory containing the directories to delete
- *array* **\$entries** list of subdirectories to delete, keyed by subdirectory name
- **\$directories**

workhorse function that actually removes directories

This routine first deletes the files "index.html" and "THUMBNAIL_PREFIX*" in the directory to remove and subsequently the directory itself. This is more or less the reverse of the mkdir function (see [task add directory\(\)](#)) but with a twist: we consider any remaining thumbnails as trash and we will happily delete those without further ado.

If anything goes wrong, the routine returns FALSE and some details are written to the logfile. Note that if the directory somehow contains symlinks or devices or named pipes we bail out: we cannot handle those kinds of directory entries.

Note that the routine is able to delete an array of directories, even though it is currently called/used with only a single entry. We don't want to make it too easy to remove many directories at once (an attempt to protect the user against herself).

- **Uses \$CFG**

bool function FileManager::delete_files(\$path, \$entries) [*line 1544*]

Function Parameters:

- *string* **\$path** the directory containing the files to delete
- *array* **\$entries** list of files to delete

workhorse function that actually deletes files, and possibly the corresponding thumbnails

This routine deletes the files specified in the array \$entries from directory \$path. If a thumbnail-file exists (ie. a file with a similar name but with the THUMBNAIL_PREFIX prepended), it is deleted too.

- **Uses \$CFG**

array function FileManager::explode_path(\$path) [*line 1735*]

Function Parameters:

- *string* **\$path** the path to split

shorthand for splitting a path into an array with path components

This routine splits \$path into components. Path components are supposed to be delimited with a forward slash '/', but if somehow a backslash '\' is encountered, it is translated to a forward slash first. This means that it is impossible to have backslashes as part of a path name, even though the underlying filesystem would happily accept a component (filename or directoryname) with an embedded backslash.

string function FileManager::file_url(\$path) [*line 2704*]

Function Parameters:

- *string* **\$path** the name of the file including path

construct a url that links to a file via /file.php

This constructs a URL that links to a file, either

```
/file.php/path/to/file.txt
```

or

```
/file.php?file=/path/to/file.txt
```

depending on the global setting for proxy-friendly urls. Note that we try to make the link as short as possible, eg. by omitting the http:// part if possible (see \$CFG->www_short).

Note: If \$path contains special characters such as '%' or '#' these are encoded via rawurlencode(). However, the path delimiter '/' (ascii 0x2F) is changed back to a genuine '/' to make \$path look like a path.

array function FileManager::get_dialogdef_add_directory() [line 2384]

construct a dialog definition for adding a subdirectory

this constructs an array which defines the dialog

array function FileManager::get_dialogdef_add_files([\$num_files = 1]) [line 2324]

Function Parameters:

- **int \$num_files** the maximum number of file upload fields to add to the dialog (default 1)

construct a dialog definition for adding (uploading) files

this constructs an array which defines a file(s) upload dialog. Note that we make a subtle difference between a single-file upload and a multifile upload: I think it looks stupid to start numbering a list of files to upload when there is in fact only a list of exactly 1 file(s). The cost is minimal: two extra strings in the translation file.

By using the fieldname 'filename[]', ie. a name ending in '[]' we tell PHP to store the uploaded file information in an array rather than in \$num_files separate entries in the global \$_FILES[] array. This has a huge advantage that any 'multiple' file uploads, ie. a comma-delimited list of filenames as supported by many browsers since Firefox 3.6 in a single file upload widget, all end up in max 5 arrays. In other words: we can work with the 0-based arrays \$_FILES['filename']['name'], \$_FILES['filename']['type'], \$_FILES['filename']['tmp_name'], \$_FILES['filename']['error'] and \$_FILES['filename']['size']. This is independent from the actual number of file input elements in the dialog. It requires setting the property 'multiple' to TRUE though. (Multiple upload feature was added July 2014)

- **Uses \$CFG**

array function FileManager::get_dialogdef_confirm_delete_directory() [line 2448]

construct a dialog definition for removing/deleting a subdirectory

this constructs an array which defines the confirmation dialog

array function FileManager::get_dialogdef_confirm_delete_files(\$entries_to_delete) [*line 2412*]

Function Parameters:

- *array* **\$entries_to_delete** contains information about the files to delete

construct a dialog definition for removing/deleting files

this constructs an array which defines the confirmation dialog

array function FileManager::get_entries(\$path) [*line 1967*]

Function Parameters:

- *string* **\$path** the directory to list, e.g. '/areas/exemplum' or '/groups/faculty'

generate a list of selected files and subdirectories in \$path

This creates an array containing a (filtered) listing of the directory \$path, keyed by filename. we items are suppressed:

- current directory '.'
- parent directory '..'
- index.html if it has size 0 (used to 'protect' directory against prying eyes)
- THUMBNAIL_PREFIX* the thumbnails of images
- symbolic links

- **Uses** \$CFG;

array function FileManager::get_entries_areas() [*line 1824*]

generate a list of (virtual) directories for areas the user can access

This generates a list of (virtual) area directories for which the user has access permissions. The list is ordered based on the sort order (in the areas table).

- **Uses** \$USER;
- **Uses** \$CFG;

array function FileManager::get_entries_groups() [line 1860]

generate a list of (virtual) directories for groups the user can access

This generates a list of (virtual) group directories for which the user has access permissions. The list is ordered by groupname.

- **Uses** \$USER;
- **Uses** \$CFG;

array function FileManager::get_entries_root() [line 1752]

generate a list of (virtual) directories at the root level

This generates a list of up to 4 'directories' which are equivalent to 'My Files', 'Areas', 'Groups' and 'Users'. Permissions and group memberships are taken into account, i.e. if a user has no group memberships (and is not an account manager), the 'Groups' directory is suppressed.

- **Uses** \$USER
- **Uses** \$CFG

array function FileManager::get_entries_users() [line 1908]

generate a list of (virtual) directories for users this user can access

This generates a list of (virtual) user directories for which this user has access permissions. The list is ordered by full name.

- **Uses** \$USER;
- **Uses** \$CFG;

int function FileManager::get_max_file_uploads() [*line 3079*]

maximum number of files accepted in a single upload (since PHP 5.2.12)

This retrieves the INI-file setting for the maximum number of files that can be uploaded in a single go (PHP 5.2.12+) OR an educated guess equal to the default value in PHP 5.2.12 (which happens to be 20). In the latter case this limit isn't actually a limit but we use it to inform the user about limits. Most users will probably have upgraded to a PHP-version newer than 5.2.12 anyway making this a moot issue...

bool function FileManager::has_allowed_extension(\$filename, &\$extensions) [*line 3033*]

Function Parameters:

- *string* **\$filename** the filename to examine
- *bool|array* **&\$extensions** array with allowable extensions or FALSE for none or TRUE for all

see if the filename extension is allowed

Note that an 'empty' extension could be acceptable.

string function FileManager::human_readable_size(\$size) [*line 2099*]

Function Parameters:

- *int* **\$size** value to convert

convert an integer filesize to a human readable form

This routine displays a file size in bytes, kilobytes, megabytes or gigabytes or bytes with space-delimited groups of 3 digits depending on the size. No decimals are used.

string function FileManager::sanitise_filetype(\$path, \$name, \$type, \$mimetype) [*line 2599*]

Function Parameters:

- *string* **\$path** full path to the actual file (from \$_FILES[\$i]['tmp_name'])

- *string* **\$name** the requested name of the file to examine (from \$_FILES[\$i]['name'])
- *string* **\$type** the suggested filetype of the file (from \$_FILES[\$i]['type'])
- *string* **\$mimetype** actual mimetype based on \$path

try to make sure that the extension of file \$name makes sense or matches the actual filetype

this checks or changes the \$name of the file in line with the mimetype of the actual file (as established by get_mimetype()).

The reason to do this is to make it harder to 'smuggle in' files with deceptive filenames/extensions. Quite often the extension is used to determine the type of the file, even by browsers that should know better. By uploading a malicious .PDF using an innocuous extension like .TXT, a browser may be tricked into rendering that .PDF inline. By changing the extension from .TXT to .PDF we can mitigate that risk, at least a little bit. (People somehow trust an extension even though they should know better and file(1) says so...)

Strategy is as follows. If the mimetype that corresponds to the extension matches the actual mimetype determined via get_mimetype(), we can simply allow the name provided.

If there is a difference, we try to find an extension that maps to the same mimetype as that of the actual file. IOW: we put more trust in the mimetype of the actual file than we do in the mimetype suggested by the extension.

If no suitable extension is available (based on the calculator mimetype) we simply use 'bin'. This changes the filename and the caller will probably reject this upload anyway because of this.

bool function FileManager::save_uploaded_file(\$full_source_path, \$target_directory, \$target_name, \$resize) [*line 2781*]

Function Parameters:

- *string* **\$full_source_path** absolute path of the uploaded file
- *string* **\$target_directory** the working directory (relative to \$CFG->datadir)
- *string* **\$target_name** the name of the image file (including extension if any)
- *bool* **\$resize** if TRUE, resize image files on the fly, move uploaded file otherwise

move the uploaded file in place or perhaps resize it first (supported images only) + create thumb

this routine performs the actual adding/moving of the uploaded file to its final destination. If the file is NOT a supported image, the effect is the same as that of the standard function `move_uploaded_file()`: the file is moved from the temporary location (e.g. `/tmp/phpAE45Rg`) to the destination within the `$CFG->datadir` (e.g. `/areas/exemplum/sample.txt`).

If the file happens to be a supported graphical file, e.g. a .GIF, a .JPG or a .PNG-file, the story is different. In that case we attempt to create a thumbnail of the source file first. The dimensions of this thumbnail are determined by a parameter in the global `$CFG`, e.g. a box of 100x100 pixels. After that and depending on the flag `$resize`, we create another copy of the sourcefile, but with a different box (say 800x800), again based on a parameter in the global `$CFG`.

If somehow the scaling appears to fail, we eventually resort to moving the file anyway in order to not leave the user empty-handed after uploading the file.

If everything fails, we return FALSE, TRUE on success.

The strategy is as follows. First we determine whether this is actually a graphics file and whether GD is available. Then we decide if scaling is needed at all. If the file is of one of the currently supported image formats, we load the file, resample it and write the resulting (smaller) image to a file which name is prepended with the thumbnail prefix. We then either move to original file (depending on the existing dimensions and the `$resize` flag) or create a bigger smaller version.

The following cases are possible (assuming `dimension=100` for thumbs, 800 for resized images)

- A. `source dimension <= 100`: move the source file to target
- B. `100 < source dimension <= 800` load the image, create a thumbnail, move the source file to target
- C. `800 < source dimension && $resize == FALSE` load the image, create a thumbnail, move the source file to target
- D. `800 < source dimension && $resize == TRUE` load the image, create a thumbnail, create resized a target (and leave source file untouched)

Design considerations:

- we use square boxex (of `$thumb_dimension` or `$resize_dimension`) to fit the images
- we do not create thumbnails for images smaller than that square box
- we do not create resized images if they are already smaller than `$resize_dimension`
- the aspect ratio is preserved
- we use default quality settings for write jpeg and png thumbnails
- all errors are logged, successes are logged to `WLOG_DEBUG`
- this routine relies on GD
- we try to extend our stay with `set_time_limit()` every time a thumbnail is created because image processing is quite time-consuming

Note that we only use GD to create thumbnails. I did consider Imagick, but eventually I decided against it because it appears that support for that extension requires PHP 5.1.3. Perhaps we can add support in a later version of the FileManager.

void function FileManager::show_breadcrumbs(\$path) [line 1375]

Function Parameters:

- *string* **\$path** the directory path to show

display a clickable path to the directory \$path

void function FileManager::show_dialog_add_files(\$dialogdef, \$href, \$path) [line 1453]

Function Parameters:

- *array* **\$dialogdef** defines the upload dialog
- *string* **\$href** the target for POSTing data
- *string* **\$path** is the target directory

display the file upload dialog

this displays the file upload dialog, including the message which reminds the user of the limits that are currently in place (in bytes).

- **TODO** should we take the locale into account formatting numbers (thousands separator)?

dialog function FileManager::show_dialog_confirm_delete_directory(\$dialogdef, \$path, \$entries_to_delete) [line 1514]

Function Parameters:

- *array* **\$dialogdef** defines the confirmation dialog
- *string* **\$path** the working directory
- *array* **\$entries_to_delete** an array with directory entries identifying the files to delete

show a dialog that ask the user to confirm the removal of a directory

Show the first directory from \$entries_to_delete and ask the user to confirm with [Delete] button or to cancel with [Cancel] button. The name of the directory to delete is part of the href rather than a POSTed field. We only allow a single directory to be removed at a time.

dialog function FileManager::show_dialog_confirm_delete_files(\$dialogdef, \$path, \$entries_to_delete) [*line 1480*]
Function Parameters:

- *array* **\$dialogdef** defines the confirmation dialog
- *string* **\$path** the working directory
- *array* **\$entries_to_delete** an array with directory entries identifying the files to delete

show a dialog that ask the user to confirm a mass file delete

Show a list of files from \$entries_to_delete and ask the user to confirm with [Delete] button or to cancel with [Cancel] button. The names of the files to delete are communicated via hidden fields. Once the form is submitted the data is validated against the existing files in the directory \$path.

- **Usedby** [FileManager::task_remove_file\(\)](#)
- **Usedby** [FileManager::task_remove_multiple_files\(\)](#)

void function FileManager::show_directories(&\$entries, \$parent) [*line 1333*]
Function Parameters:

- *array* **&\$entries** ready to use data describing all subdirectories to show
- *bool/string* **\$parent** suppress link to parent if FALSE otherwise path of parent

output a simple list of directories (for navigation only)

This outputs a simple list of subdirectories based on information in the array \$entries. The subdirectories can not be deleted and no files or subdirectories can be added. (because this is either '/', 'Areas', 'Groups' or 'Users')

- **Used by** [FileManager::show_list\(\)](#)
- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

void function FileManager::show_directories_and_files(\$path, [\$show_thumbnails = TRUE]) [*line 971*]

Function Parameters:

- *string* **\$path** the directory to display
- *bool* **\$show_thumbnails** if TRUE files are displayed as thumbnails, table rows otherwise

display a list of subdirectories and files in directory \$path

This long routine displays the following items to the user

- (optional) navigation link to add (upload) a file
- (optional) navigation link to add (create) a directory
- a 4, 5 or 6 column table with
 - . navigation link to the parent directory
 - . 0, 1 or more rows with delete and navigation links to subdirectories (if any)
 - . 0, 1 or more rows with a checkbox and delete and preview links to files (if any)
 - . (optional) a 'select all' checkbox
- (optional) Delete-button to mass-delete files

The table can be ordered in various ways: by name, by size and by date, either ascending or descending. Clicking the relevant column header yields another sort order. This toggles between ascending and descending. Default sort order is by name ascending.

The checkbox 'select all' works with Javascript in the most simple way: an ad-hoc script connected to the onclick attribute of the select all checkbox. However, the select all checkbox itself is rendered via Javascript. The effect is that this feature is only available if Javascript is enabled in the browser. If it isn't, no select all is visible so it can not distract the user. This is part of the attempt to make this CMS usable even without Javascript.

If the flag \$show_thumbnails is set we display file entries as thumbnails. This is done mostly to cater for the visual interactive selection of images from FCK Editor.

- **TODO** This routine is way too long, it should be split up into smaller subroutines
- **Usedby** [FileManager::show_list\(\)](#)
- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER
- **Uses** \$CFG

output function FileManager::show_file_as_thumbnail(\$directory, \$entry, \$delete_file, \$index, [\$m = ""]) [line 2920]

Function Parameters:

- *string* **\$directory** the current working directory (necessary to construct (full) paths)
- *array* **\$entry** information about the file to show, see [get_entries\(\)](#) for the format
- *bool* **\$delete_file** if TRUE, user is allowed to delete the file (used for creating delete icon)
- *int* **\$index** a counter used to generate a unique field name for the checkbox
- *string* **\$m** optional margin for better code readability

show a thumbnail of a single (image) file perhaps including clickable links for use in FCK Editor

This constructs a single clickable image with either a selection of the file (for FCK Editor, in file/image browser mode) or a link to the file preview. If a file is not an image or otherwise no suitable thumbnail is found, a large question mark is displayed (unknown.gif). Otherwise the existing thumbnail is shown, maintaining the original aspect ratio. Either way the image is scaled to the currently specified thumbail dimension so the image fits the corresponding DIV-tag.

The strategy for finding a thumbnail is as follows: - is the file to show an image at all? If not, show unknown.gif - if the file zz_thumb_{filename.ext} exists, use that, otherwise - if not AND the original file is smaller than a thumbailm use the original file, otherwise - use 'unknown.gif' after all.

If the flag \$delete_file is set, we also generate a checkbox and a delete icon. This means that even in file/image browser mode files can be deleted by the user. In fact the file/image browser is basically the same old filemanager.

- **Uses** \$WAS_SCRIPT_NAME

- **Uses** \$CFG

void function FileManager::show_list(\$path) [*line 906*]

Function Parameters:

- *string* **\$path** the (virtual) path to the directory to list

display a list of directories and files in \$path

This yields a list of directories (for \$path is '/', '/areas', '/groups' or '/users') or a list of directories and files (\$path is anything else). In the latter case, if the current user has sufficient permissions, various additional links are added such as upload a file or create directory. The actual work is done in two separate workhorses.

- **Uses** [FileManager::show_directories_and_files\(\)](#)
- **Uses** [FileManager::show_directories\(\)](#)

void function FileManager::show_menu([\$current_path = ""]) [*line 1418*]

Function Parameters:

- *string* **\$current_path** indicator for highlighting the current directory subtree

show a menu that is equivalent with the root directory

void function FileManager::sort_entries(&\$entries, \$sort, \$sortorder) [*line 2142*]

Function Parameters:

- *array* **&\$entries** array with directory entries
- *int* **\$sortorder**
- **\$sort**

sort directory entries

- **TODO** it is a pity I cannot reference `$this->sort` from within the 6 cmp-functions...

void function FileManager::task_add_directory() [line 677]

create a new subdirectory

This routine either shows a dialog where the user can specify the name of a new directory to add OR processes the dialog.

In case of directory name too short, already exists, etc. the user is returned to the dialog to try again. If all goes well the new directory is created and at the same time the empty file 'index.html' is created to "protect" the directory from prying eyes.

void function FileManager::task_add_file() [line 472]

add one or more new files to a directory

This routine either shows a dialog where the user can specify the names of one or more (maximum `$CFG->upload_max_files`) files OR processes the dialog.

Various checks are performed before the files are actually saved, e.g. checks for viruses (via ClamAV), resolve name clashes, allowed filetypes and extensions, etc. This is all done in a separate worker routine.

void function FileManager::task_change_directory() [line 221]

make another directory the current (working) directory and optionally change the sort order

This changes the current working directory to the user-supplied path (after thorough validation, naturally). The new current directory is stored in `$this->current_directory` and via that reference in the `$_SESSION` array, for future reference. If a valid directory was specified, we also take a look at the optional sort order parameter and set the sort order of the directory listing accordingly.

After (perhaps) changing the current directory, and perhaps changing the sort order, the contents of that directory is displayed via [task_list_directory\(\)](#).

void function FileManager::task_list_directory() [line 195]

show a directory listing of the current working directory and links to add/delete files/directories etc.

This is the main routine to show a list of subdirectories and files in the current working

directory (\$_SESSION['current_directory']).

void function FileManager::task_preview_file() [line 240]

preview a file via file.php

After validation of the specified path, the user is redirected to [file.php](#) in order to show the selected file.

void function FileManager::task_remove_directory() [line 387]

show a confirmation dialog for removing a single directory OR actually removes a directory

This shows a confirmation dialog for removing of a single directory OR actually removes a directory.

void function FileManager::task_remove_file() [line 344]

show a confirmation dialog for deleting a single file

This shows a confirmation dialog for deletion of a single file. We reuse the code for deletion of multiple files, see [task_remove_multiple_files\(\)](#).

- **Uses** [FileManager::show_dialog_confirm_delete_files\(\)](#)

void function FileManager::task_remove_multiple_files() [line 261]

show confirmation dialog for multiple file delete OR perform actual file delete

this routine either shows a list of files to be deleted, asking the user for confirmation or actually deletes the specified files if the user did confirm the delete. We bail out if the user pressed the cancel button in the confirmation dialog. The real work is done in workhorse routines in order to combine the single-file-delete and the batch-delete into a single confirmation routine. For actual deletion, however, we always return here and not in the single file delete (see `{link task_remove_file()}`).

- **Uses** [FileManager::show_dialog_confirm_delete_files\(\)](#)

string function FileManager::unique_filename(\$directory, \$name) [*line 2654*]

Function Parameters:

- *string* **\$directory** the target directory for the file upload
- *string* **\$name** the (sanitised) name of the file

construct a unique filename taking existing files into account

this constructs a filename that is unique within the target directory. If a file of the name \$name already exists, a new name is constructed from the basename of \$name, an integer sequence number and the extension of \$name.

There is no guarantee that this name will stay unique between the moment we test for it and the moment the file is actually moved into place (a classical race condition). However, the chance that this will be happening is small enough I guess.

- **TODO** Should we take care of the race condition in this routine? Should we already create an empty file or is that clutter?
- **Uses** \$CFG

string|bool function FileManager::valid_path(\$path) [*line 811*]

Function Parameters:

- *string* **\$path** the path (file or directory) to check, relative to \$CFG->datadir

access control and validation for selected directory or file

This routine checks to see if the current user has access to the specified file or directory. If not, FALSE is returned, otherwise the valid path is returned, with a starting slash. If \$path doesn't start with a slash a slash is assumed nevertheless. The path is relative to \$CFG->datadir.

It is not allowed to reference parent directories in the path. This prevents tricks like '../../etc/passwd' (leaking the system passwd file) or '/users/foo/./bar' (access to bar's userdata with permissions for just foo's files). Furthermore, symbolic links are NOT acceptable as part of the path, i.e. symlinks like '/users/foo/etc -> /etc' or '/users/foo/passwd -> /etc/passwd' are considered invalid.

Required permissions for access are:

- areas: \$USER must have the admin_pagemanager permissions for that area.
- groups: \$USER must be a member of that group OR \$USER must have access to the Account Manager.
- users: \$USER must be that user OR \$USER must have access to the Account Manager.

- **TODO** the check on '../' is inconclusive if the \$path is encoded in UTF-8: the overlong sequence 2F C0 AE 2E 2F eventually yields 2F 2E 2E 2F or '../'. Reference: RFC3629 section 10.
- **Uses** \$USER
- **Uses** \$CFG

int function FileManager::virusscan(\$path, [\$name = ""]) [*line 2487*]

Function Parameters:

- *string* **\$path** the path of the file to scan
- *string* **\$name** the name of the file as provided by the uploader (from \$_FILES)

scan a file for viruses

this scans \$path for viruses, returns 0 if file considered clean, 1 for infected file, or 2 if something else went wrong.

If the flag \$CFG->clamscan_mandatory is set, we consider the file infected if we are not able to run the virus scanner (better safe than sorry). However, if no virusscanner is configured at all (\$CFG->clamscan_path is empty), we indicate a 'clean' file even though we did not scan it. Rationale: it doesn't make sense to make scanning mandatory and at the same time NOT configuring a scanner at all.

If scanning succeeds and a virus is found we send an alert to the website owner address (or the reply-to-address) immediately. Furthermore everything is logged.

- **TODO** This routine is quite *nix-centric. I'm not sure how this would work other server platforms. Should we do something about that?

- **TODO** maybe use MIME for sending alert if not 7bit message?
- **Uses** \$USER
- **Uses** \$CFG

string function FileManager::vname(\$path) [line 1664]

Function Parameters:

- *string* **\$path** the path to examine

construct the (possibly translated) name of the last directory in the path

This examines \$path and returns a string with the last directory component. There are a few special cases:

- the empty string indicates the root directory
- /users/<userpath> maps to a (translated) string t('filemanager_personal')
- /areas maps to a (translated) string t('filemanager_areas')
- /groups maps to a (translated) string t('filemanager_groups')
- /users maps to a (translated) string t('filemanager_users')

All other variations yield the last component in the list of components.

string function FileManager::vpath(\$path) [line 2032]

Function Parameters:

- *string* **\$path** the path to translate

translate a path to the corresponding virtual path

This translates a path like '/users/webmaster/foo' into 'My Files/foo' and '' into 'All Files', etc. The result of the translation is cached in \$this->vpaths, for future reference.

Class GroupManager

[line 37]

Group management

- **Package** wascore
- **TODO** Perhaps this class should be merged with the UserManager class because there is a lot of overlap. Mmmmm.... maybe in a future refactoring operation.

GroupManager::\$groups

array = array() [line 48]

- **Var** used to cache group records keyed by group_id

GroupManager::\$group_capacity_records

array|null = NULL [line 42]

- **Var** caches the list of group-capacity-combinations

GroupManager::\$output

object|null = NULL [line 39]

- **Var** collects the html output

GroupManager::\$show_parent_menu

bool = FALSE [line 45]

- **Var** if TRUE the calling routing is allowed to use the menu area (e.g. show account mgr menu)

Constructor *void* function GroupManager::GroupManager(&\$output) [*line 56*]

Function Parameters:

- *object* **&\$output** collects the html output

construct a GroupManager object

This initialises the GroupManager and also dispatches the task to do.

bool function GroupManager::add_group_capacity(\$group_id, \$capacity_code, \$sort_order) [*line 1723*]

Function Parameters:

- *int* **\$group_id** group of interest
- *int* **\$capacity_code** the capacity
- *int* **\$sort_order**

add a group/capacity and corresponding acl to the database

array|bool function GroupManager::areas_expand_collapse(\$areas_open, \$area_id) [*line 1870*]

Function Parameters:

- *array|bool* **\$areas_open** current state of indicator(s) for 'open' and 'closed' areas
- *int|null* **\$area_id** the area to expand/collapse or NULL if nothing needs to be done

manipulate the current state if indicator(s) for 'open' and 'closed' areas

this manipulates the current state of 'open' and 'closed' areas in \$areas_open. If \$area_id is NULL, we don't have to do anything but simply return the current state. If \$area_id is 0 (zero), we need to toggle all areas at once (area_id = 0 implies the site level toggle) If \$area_id is an integer, it is assumed to be a valid area_id and that area should be toggled.

array function GroupManager::a_params([\$task = NULL], [\$group_id = NULL], [\$capacity_code = NULL], [\$module_id = NULL]) [*line 1587*]

Function Parameters:

- *string|null* **\$task** the next task to do or NULL if none
- *int|null* **\$group_id** the group of interest or NULL if none
- *int|null* **\$capacity_code** the capacity of interest or NULL if none
- *int|null* **\$module_id** the module of interest or NULL if none

shorthand for the anchor parameters that lead to the group manager

int|bool function GroupManager::calc_acl_id(\$group_id, \$capacity_code) [*line 1668*]

Function Parameters:

- *int* **\$group_id** the group to examine
- *int* **\$capacity_code** the capacity to examine

retrieve the acl_id for a particular group/capacity from the database

void function GroupManager::capacity_admin() [*line 1033*]

show a dialog for modifying admin permissions for a group/capacity

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

void function GroupManager::capacity_intranet() [*line 996*]

show a dialog for modifying intranet permissions for a group/capacity

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

void function GroupManager::capacity_overview() [line 818]

display an overview of all members of a group with a particular capacity

this constructs a clickable list of users that are associated with a particular combination of group_id and capacity_code. The name of the user is a link to the usermanager for that user. Users are sorted by name.

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$DB

void function GroupManager::capacity_pagemanager() [line 1070]

show a dialog for modifying page manager permissions for a group/capacity

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

void function GroupManager::capacity_save() [line 874]

save data from a dialog for a group/capacity

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

bool function GroupManager::delete_group_capacities_records(\$group_id) [line 1779]

Function Parameters:

- *int* **\$group_id** the group to delete

actually remove a group and all associated records

this actually deletes the group \$group_id and associated records, in a specific order, to satisfy the FK constraints. ACL's are deleted last if there are any at all.

Note This routine looks a lot like the corresponding one in the UserManager. However, we don't know in advance how many ACLs are associated with this group whereas a user record always has exactly 1 ACL. This explains the logic in step 1 below.

- **TODO** since multiple tables are involved, shouldn't we use transaction/rollback/commit? Q: How well is MySQL suited for transactions? A: Mmmmm.... Which version? Which storage engine?

array function GroupManager::get_dialogdef_add_group() [*line 1383*]

construct the add group dialog

array|bool function GroupManager::get_dialogdef_edit_group(\$group_id) [*line 1444*]

Function Parameters:

- *int* **\$group_id** the group that will be edited

construct the edit group dialog

string function GroupManager::get_groupname(\$group_id) [*line 1657*]

Function Parameters:

- *int* **\$group_id** the group of interest

shorthand to get the name of a group

array function GroupManager::get_group_capacity_names(\$group_id, [\$capacity_code = 0]) [*line 1706*]

Function Parameters:

- *int* **\$group_id** identifies the group of interest
- *int* **\$capacity_code** identifies the capacity (optional)

shortcut to retrieve the name and full name of the selected group and optionally a capacity name

array|bool function GroupManager::get_group_capacity_records([\$force = FALSE]) [*line 1559*]

Function Parameters:

- **\$force**

return an array of group-capacity records (possibly buffered)

- **Uses \$DB**

bool|array function GroupManager::get_group_record(\$group_id, [\$forced = FALSE]) [*line 1841*]

Function Parameters:

- *int* **\$group_id** identifies the group record
- *bool* **\$forced** if TRUE unconditionally fetch the record from the database

retrieve a single group's record possibly from the cache

string function GroupManager::get_icon_delete(\$group_id) [*line 1612*]

Function Parameters:

- *int* **\$group_id** the group to delete

construct a clickable icon to delete this group

- **Uses \$WAS_SCRIPT_NAME**
- **Uses \$USER**
- **Uses \$CFG**

string function GroupManager::get_icon_edit(\$group_id) [*line 1636*]

Function Parameters:

- *int* **\$group_id** the group to edit

construct a clickable icon to edit the properties of this group

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER
- **Uses** \$CFG

array function GroupManager::get_options_capacities() [*line 1541*]

construct a simple option list with all available capacity names keyed by capacity code

void function GroupManager::groups_overview() [*line 133*]

display list of existing groups and an option to add a group

this constructs the heart of the group manager: a link to add a group followed by a list of links for all existing groups and additional links per capacity per group.

This list of groups is ordered as follows. All active groups come first, the inactive groups follow. The sort order is based on the (short) name of the group.

Example:

Add a group

[D] [E] faculty (Member, Principal)

[D] [E] grade12 (Pupil, Teacher)

...

[D] [E] zebra (Member, Project lead)

[D] [E] aardvark (inactive)

[D] [E] grade45 (inactive)

Note that both the links '[E]' and 'faculty' lead to edit of group properties The links 'Member' and 'Principal' lead to the group-capacity overview screen The link '[D]' leads to a group delete confirmation screen

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER
- **Uses** \$CFG

void function GroupManager::group_add() [line 252]

present 'add group' dialog where the user can enter minimal properties for a new group

this displays a dialog where the user can enter the minimal necessary properties of a new group. These properties are:

- name (e.g. 'grade7')
- full name (e.g. 'Pupils of grade 7')
- the active flag
- the allowable capacities for this group (e.g. 'Pupil' and 'Teacher')

Other properties (if any) will be set to default values and can be edited later on by editing the group.

The new group is saved via performing the task TASK_GROUP_SAVE_NEW

- **Uses** \$WAS_SCRIPT_NAME

void function GroupManager::group_delete() [line 668]

delete a group after confirmation

this either presents a confirmation dialog to the user OR deletes a group with associated capacities and acs.

Note that this routine could have been split into two routines, with the first one displaying the confirmation dialog and the second one 'saving the changes'. However, I think it is counter-intuitive to perform a deletion of data under the name of 'saving'. So, I decided to use the same routine for both displaying the dialog and acting on the dialog.

- **TODO** since multiple tables are involved, shouldn't we use transaction/rollback/commit? Q:

How well is MySQL suited for transactions? A: Mmmmm.... Which version? Which storage engine?

void function GroupManager::group_edit([\$group_id = NULL]) [line 415]

Function Parameters:

- *int \$group_id* which group do we want to edit?

show a dialog with the basic properties of a group

- **Uses \$WAS_SCRIPT_NAME**

void function GroupManager::group_save() [line 448]

save an edited group to the database, including adding/modifying/deleting group/capacity-records

Note: no error checking when inserting new capacity because we more or less know that that capacity does not exist already or it would have been in the array already. (But what if there are more than GROUPMANAGER_MAX_CAPACITIES in the database? Mmmm....

- **Uses \$WAS_SCRIPT_NAME;**

data function GroupManager::group_savenew() [line 286]

save a new group to the database

this saves a new group to the database. This quite a complex task because of the number of tables involved.

First we have the table 'groups' which stores the basic group information. Then there is the table 'groups_capacities'. For every combination of group and capacity requested by the user a record must be added to this table. Then there is also a separate acl for every group_capacity, so there.

The strategy should be something like this. `new_group_id = insert_new_group_into_groups()`
for all `GROUPMANAGER_MAX_CAPACITIES` do if `capacity != CAPACITY_NONE`
&& `capacity_not_added_yet()` prepare_new_acl_record(); new_acl_id =

```
insert_new_acl_in_acls();           prepare_new_groups_capacities_record();
insert_new_group_capacity_in_table()
```

- **TODO** maybe we should find a more elegant way to check a field for uniqueness
- **TODO** should we delete the datadirectory if something goes wrong?

bool function GroupManager::has_job_permission(\$group_id, \$capacity_code, \$job) [*line 1258*]

Function Parameters:

- *int* **\$group_id** group to check
- *int* **\$capacity_code** capacity of this group to check
- *int* **\$job** job a bitmask indicating a particular job

determine whether a group/capacity has permissions for a particular job

this determines whether this group/capacity has permissions to access the specified job, e.g. do they have access to the page manager. If so, we can display the menu option, otherwise we can suppress it and keep the menu clean(er).

void function GroupManager::show_breadcrumbs_addgroup() [*line 1353*]

display breadcrumb trail that leads to the add new group dialog

- **Uses** \$WAS_SCRIPT_NAME;

void function GroupManager::show_breadcrumbs_group() [*line 1273*]

display breadcrumb trail that leads to groups overview screen

- **Uses** \$WAS_SCRIPT_NAME;

void function GroupManager::show_breadcrumbs_groupcapacity(\$group_id, \$capacity_code) [line 1139]

Function Parameters:

- **\$group_id**
- **\$capacity_code**

display breadcrumb trail that leads to group capacity overview screen

- **Uses \$WAS_SCRIPT_NAME;**

void function GroupManager::show_menu_group(\$group_id, [\$current_task = NULL], [\$current_capacity_code = NULL]) [line 1301]

Function Parameters:

- *int* **\$group_id** the group of interest
- *string|null* **\$current_task** the name of the task to emphasise in the menu (underlined)
- *string|null* **\$current_capacity_code** the name of the capacity to emphasise in the menu (underlined)

show a menu for a group including links to the group's capacity overview screens

- **Uses \$WAS_SCRIPT_NAME;**

void function GroupManager::show_menu_groupcapacity(\$group_id, \$capacity_code, [\$current_task = NULL], [\$current_module_id = NULL], \$current_capacity_code) [line 1178]

Function Parameters:

- *int* **\$group_id** the group of interest

- *int* **\$capacity_code** identifies the capacity to manage
- *string|null* **\$current_task** the name of the task to emphasise in the menu (underlined)
- *string|null* **\$current_capacity_code** the name of the capacity to emphasise in the menu (underlined)
- **\$current_module_id**

show a menu for a group capacity with options to modify privileges, etc.
etc.

- **Uses** \$WAS_SCRIPT_NAME;

void function GroupManager::show_parent_menu() [*line 92*]

bool function GroupManager::valid_group_capacity(\$group_id, \$capacity_code) [*line 1687*]

Function Parameters:

- *int* **\$group_id** the group to examine
- *int* **\$capacity_code** the capacity to examine

shorthand to test the validity of a particular group/capacity

Class Language

[*line 33*]

Translations of messages in different languages

- **Package** wascore

Language::\$default_domain

string = [line 41]

- **Var** \$default_domain the text domain to use if none is specified

Language::\$languages

array = array() [line 38]

- **Var** \$languages a cached list of all language records

Language::\$phrases

array = array() [line 35]

- **Var** \$phrases a cache of translated phrases

Constructor *void* function Language::Language([\$default_domain = ""]) [line 51]

Function Parameters:

- *string* **\$default_domain** used when no domain is specified when requesting a translation

constructor

Set up the instance of this class. If no default domain is specified, 'was' is used. We always read the current list of all languages into core, for future reference.

array function Language::get_active_language_names() [line 109]

return an array with active languages and language names

this returns an array with language_key => language_name pairs, one entry per active

language, ordered by language name. This array can be used in language picklists or to translate a language key to readable form. Note that we use the name of a language expressed in the language itself.

string function Language::get_current_language() [*line 144*]

determine the default language to use for translation of phrases

This routine determines which language to use for prompts and messages if not specified explicitly in calls to `$this->get_phrase()`. There are various ways in which a language can be determined. Here's the list, in order of significance:

- `$_GET['language']`
- `$_SESSION['language_key']`
- `$USER->language_key`
- `$CFG->language_key`
- constant value 'en' (the native language)

Note that all languages are validated against the list of valid and active languages as collected in `$this->languages`. If a language is NOT valid, the next test is tried. If all else fails we return 'en' for English, which is the native language and which should always be valid.

- **Uses** `$USER`;
- **Uses** `$CFG`

array function Language::get_filenames_to_try(`$full_domain`, `$location_hint`, `$language_key`) [*line 453*]

Function Parameters:

- *string* **`$full_domain`** indicates the text domain including optional module/theme/addon prefix
- *string* **`$location_hint`** hints at a location of language file(s)
- *string* **`$language_key`** target language

calculate an ordered list of filenames to try for translation of phrases

WAS uses a separate language file for every text domain; basically the name of the text domain is the name of the file without the .php-extension. However, in order to prevent name clashes, modules and themes and addons have their own prefix: 'm_' for modules and 't_' for themes and 'a_' for addons.

The language translations for the installer are based on more or less the same trick: the prefix

string function Language::get_phrase(\$phrase_key, [\$full_domain = "], [\$replace = "], [\$location_hint = "], [\$language_key = "]) [*line 317*]

Function Parameters:

- *string* **\$phrase_key** indicates the phrase that needs to be translated
- *string* **\$full_domain** (optional) indicates the text domain (perhaps with a prefix)
- *array* **\$replace** (optional) an assoc array with key-value-pairs to insert into the translation
- *string* **\$location_hint** (optional) hints at a directory location of language files
- *string* **\$language_key** (optional) target language

translation of phrases via a phrase key

This routine looks up the text associated with the phrase key. If no domain is specified, the domain 'was' is tried. If no valid language is specified, the current language is used. If a location hint is specified, we trust the caller knows best where to look and we try locating a translations file in that directory location first.

Note that phrases in a particular language which are found later in the search overwrite the phrases found earlier. These additional phrases (dubbed 'dialect' or 'userdefined translations') can be used to overwrite or correct existing standard ('official') translations. These dialect phrases can be stored in a file in the languages subdirectory of the data directory (writable for the web server but hopefully outside the document root) and/or in the table 'phrases' in the database.

Whether these dialect phrases are actually fetched from disk or database depends on the configuration of the language, via the boolean fields 'dialect_in_database' and 'dialect_in_file'; if the corresponding switch is not TRUE, we don't even bother to go and look, which saves time.

Finally, if a particular phrase is not found in the requested language, we recursively try the parent language(s) of the requested language until there are no more parents. After that, we go for the 'en' translation. If that fails too, we return the phrase_key itself, sandwiched between the strings '(lang) ' and ' (/lang)', where 'lang' is the requested language code. Of course this should not happen if all translations are correct. (Famous last words...)

If a translation is found, we replace all occurrences of the keys in the array 'replace' in the translation with the corresponding values. This is done via a simple search and replace rather than a printf()-like way or (shudder) with complicated regex'es.

Note that we store search results in the array \$this->phrases so we can re-use those phrases in a next call. We cache the results on a per-domain basis, based on the assumption that after the first phrase in a particular domain is requested, it is likely that more phrases in the same domain will be requested.

Note that the resulting phrases are cached using the original language as the key (in \$this->

>phrases). This means that if a phrase in say 'de' or 'fr' was not found and 'en' was used instead, the English phrases are cached in the 'de' or 'fr' branch of the static array. This saves us time on the next call because we then use the phrases in the substitute language right away instead of going to look everywhere everytime.

Translations are fetched in such a way that the user-defined translations ('dialect') prevail over the system-defined ('official') translations. However, attempts to look for a phrase in a parent language (or 'en') only add the missing translations, preserving the translations in this full_domain that were already found. Quick illustration with Dutch (nl) and English (en): search order is: \$nl_database, \$nl_userfile, \$nl_system, \$en_database, \$en_userfile, \$en_system. The 'en' translations are only used if no corresponding Dutch translation is found. However, the English 'dialect' prevails over the English 'system' translation.

Examples of typical use of this routine:

```
echo $LANGUAGE->get_phrase('username');  
display the phrase from 'was.php' in the current language
```

```
echo $LANGUAGE->get_phrase('username','login');  
display the phrase from 'login.php' in the current language
```

```
echo $LANGUAGE->get_phrase('welcome','',array('{USERNAME}' => $USER->username));  
display the phrase from was.php in the default language, substituting the variable '{USERNAME}'.
```

- **TODO** should we return an error for an invalid specific language?
- **Uses** \$LANGUAGES

array function Language::get_phrases_from_database(\$full_domain, \$language_key) [line 216]

Function Parameters:

- *string* **\$full_domain** text domain to look for
- *string* **\$language_key** the language to look for

retrieve phrases from the database for specified domain and language

array function Language::get_phrases_from_file(\$filename) [line 201]

Function Parameters:

- *string* **\$filename** which language file to include

return the \$string array after including a file

This includes the specified language file **\$filename** (if it exists) and returns the array **\$string**. This assumes that filename actually consists of lines like this:

```
...
$string['key_of_a_phrase'] = 'content of this phrase';
$string['key_with_variable'] = 'Hello, {USERNAME}.';
...
```

Because the file is included within the context of this function, the contents are added to the local array **\$string** rather than some global array. This is a feature.

Note that the included file **MUST** name the array '**\$string**' because otherwise this function will return an empty array. This means that any 'old' Site@School language files must be manipulated before they can be re-used. I'd consider this a feature too.

void function Language::reset_cache([\$language_key = "], [\$full_domain = "]) [*line 500*]

Function Parameters:

- *string* **\$language_key** the language
- *string* **\$full_domain** the language domain

remove selected entries (per language+domain, per language, or all) from cache

array function Language::retrieve_languages([\$force_reread = FALSE]) [*line 69*]

Function Parameters:

- *bool* **\$force_reread** if TRUE we always go to the database, else we try the cached version first

retrieve an array with all active languages from the database

This reads all languages from the database. If there's nothing there, we still return an array with a single element for the English language 'en', because 'en' is the native language of this program. If the language 'en' was not found, we still add it to the array. The resulting array is usually sorted by language name.

Class PageManager

[line 83]

Page Manager

This class implements the Page Manager.

All the work is directed from the constructor, so it is enough to simply instantiate a new object and let the constructor do the work. The only thing needed is an output object (see [AdminOutput](#)).

- **Package** wascore

PageManager::\$areas

null|array = NULL [line 88]

- **Var** \$areas holds all area records (for future reference) or NULL if not yet set

PageManager::\$area_id

null|int = NULL [line 94]

- **Var** \$area_id indicates which tree is stored in \$this-tree, or NULL if none yet

PageManager::\$output

object|null = NULL [line 85]

- **Var** \$output collects the html output

PageManager::\$tree

null|array = NULL [line 91]

- **Var \$tree** holds the complete tree for area \$this->area_id or NULL if not yet set

Constructor *void* function PageManager::PageManager(&\$output) *[line 119]*

Function Parameters:

- *object* **&\$output** collects the html output

construct a PageManager object (called from /program/main_admin.php)

This initialises the PageManager, checks user permissions and finally dispatches the tasks. If the specified task is not recognised, the default task TASK_TREEVIEW is executed.

Note that almost all commands act on the area contained in the SESSION-variable `current_area_id`. Also, we almost always need the tree of nodes in that area, so we read it once, `_before_` dispatching the task at hand. This means that the current tree in the current area is ALWAYS available. This means that none of the other routines should have to worry about which area or reading the tree; this information is already available in `$this->area_id` and `$this->tree`.

Note that it IS possible to perform a change of area, by specifying the new `area_id` in the command line. If this is the case, the request will not look at the 'old' `area_id` (from `$_SESSION`) but start from scratch in the 'new' area.

void function PageManager::build_cached_tree(\$area_id, [\$force = FALSE]) *[line 4103]*

Function Parameters:

- *int* **\$area_id** indicates which area to buffer if not already buffered
- *bool* **\$force** re-read of the tree for area \$area_id

construct \$this->tree for future reference

this constructs the tree of the area \$area_id so all other routines can simply use that tree

instead of passing it around again and again via function arguments.

int function PageManager::calculate_new_sort_order(&\$tree, \$area_id, \$parent_id) [*line 3907*]

Function Parameters:

- *array* **&\$tree** reference to the tree in area \$area_id
- *int* **\$area_id** the area to look at
- *int* **\$parent_id** the section where we need to make room (where a node is added/inserted)

calculate a new sort order and at the same time make room for a node

this is used to calculate a new sort order number for a node that will be added to section \$parent_id in area \$area_id. Note that this could be another area than the current working area. The reference to the tree is necessary; we can't simply use \$this->tree and \$this->area_id.

Depending on the configuration flag \$CFG->pagemanager_at_end the node is added at the end of the section or at the beginning. In the latter case, the new sort order number is always 10 and all the existing nodes are renumbered in such a way that the second node in the section (originally the first one) gets sort order 20. By not using consecutive numbers it is possible to 'insert' nodes without touching anything. This is not used but it does no harm to have a sort order in steps of 10 instead of 1. (I think the database doesn't care much when executing/interpreting the ORDER BY clause).

Note that this routine not only calculates a sort order but it also manipulates the database and moves other nodes in the section around in order to make room.

- **Uses \$DB**
- **Uses \$CFG**

int function PageManager::calculate_updated_sort_order(\$node_id, \$after_id) [*line 3997*]

Function Parameters:

- *int* **\$after_id** the node AFTER which \$node_id should be sorted (0 means: first in the section)
- **\$node_id**

calculate an updated sort order and also make space in the section for moving the node around

this calculates a new sort order for node `node_id`; the effect should be that `node_id` will sort AFTER node `after_id`. If `after_id` is 0 then `node_id` should become the first in the section.

Note that this routine not only calculates a sort order but it also manipulates the database and moves other nodes in the section around in order to make room.

There are several different cases possible: a. `$after_id == 0` b. `sort_order($after_id) < sort_order($node_id)` c. `sort_order($node_id) < sort_order($after_id)` d. `$after_id` is the last node in this section e. `$node_id == $after_id`

Case e. should not happen but if it did it would yield a no-op. Case d is very similar to case c, so much even that both cases can be combined to just one.

```
Strategy for case a. $old_sort_order = sort_order($node_id); $new_sort_order =
sort_order(first_child(parent_section($node_id))) $delta = $old_sort_order -
sort_order(prev($node_id)) SET $sort_order += $delta WHERE $new_sort_order <=
sort_order(node) <= $old_sort_order
```

In other words: `node_id` gets the `sort_order` value from the first node in the section, all nodes from the first upto position where `node_id` was originally move 'up' in such a way that the last in that range will end up with the sort order that `node_id` had originally.

```
Strategy for case b. $old_sort_order = sort_order($node_id) $new_sort_order =
sort_order(next($after_id)) $delta = $old_sort_order - sort_order(prev($node_id)) SET
$sort_order += $delta WHERE $new_sort_order <= sort_order(node) <= $old_sort_order
```

Note that a and b are also quite similar.

```
Strategy for case c. (and d.) $old_sort_order = sort_order($node_id) $new_sort_order =
sort_order($after_id) $delta = $old_sort_order - sort_order(next($node_id)) (note that this is a
negative value) SET $sort_order += $delta WHERE $old_sort_order <= sort_order(node)
<= $new_sort_order
```

By mass-updating the other nodes, we hopefully don't disturb the other nodes, even while they might be locked. So there, the lock on the node is not absolute, we will change the record behind the back of another user holding a lock. On the other hand: messing up the sort order is less messy than messing with the actual content of a node. I'll take the risk. Worst case is that two processes will both update the sort order, perhaps yielding two nodes with the same `sort_order` value. Oh well, so be it. (There is this law by Pareto, something about 80 - 20. Mmmm...)

- **TODO** Clean up this code, it is very hairy
- **Uses** \$DB

void function PageManager::calc_home_id(\$node_id, 1) [line 4080]

Function Parameters:

- *int* **\$node_id** the node of interest
- *bool/int* **1** FALSE if no default node found, the default node_id otherwise

calculate the current default node on this level

this tries to find a sibling of the node \$node_id that has the flag 'is_default' set to TRUE

bool function PageManager::delete_node(\$node_id) [line 2038]

Function Parameters:

- *int* **\$node_id** the page or the section to delete

workhorse routine for deleting a node, including children

This deletes the children (but not grandchildren) of a section and the section itself OR simply the node itself. See function for more on this design decision.

This routine actually deletes nodes from the database, but only if these nodes do not have children AND if the nodes are not readonly. Furthermore, just before the child nodes are deleted, a lock on that node is obtained. This makes sure that a node that is currently being edited by another user is not deleted under her nose. Also, we do not delete nodes that have children because that would yield orphan nodes.

Any problems with deleting children are reported in messages via \$this->output. If all children are deleted successfully, then \$node_id is deleted. Success of the whole operation is indicated by returning TRUE, otherwise FALSE.

array function PageManager::get_dialogdef_add_node(\$is_page) [line 2915]

Function Parameters:

- *bool* **\$is_page** TRUE if the dialog is for a new page, FALSE is for a new section

construct a dialog definition for adding a node (page or section)

the dialog for pages and sections are different in just a single field: the page has an extra module field.

Note that we set two default values: one for visibility and one for the default module id. For now we set the initial visibility to 2 (hidden). The default module is 1, under the assumption that the first module in the system is the one used most: a plain page. I didn't consider it worthy enough to make this defaults configurable. However, the sort order in the `get_options_modules()` doesn't guarantee that the plain page module is the first in the list, so there.

- **TODO** should we make the defaults in this routine configurable? (I'm not convinced they should)

array function PageManager::get_dialogdef_edit_advanced_node(\$node_id, \$is_page, [\$viewonly = FALSE], \$area_id) [line 3098]

Function Parameters:

- *int* **\$area_id** the area in which the node lives
- *int* **\$node_id** the node that is to be edited
- *bool* **\$is_page** TRUE if the dialog is for a page, FALSE is for a section
- *bool* **\$viewonly** if TRUE, most fields are 'dimmed' (uneditable)

construct a dialog definition for editing advanced properties of a node (page or section)

this constructs a dialog to edit the advanced properties of a node. There is a slight difference between pages and sections: a section can have neither the 'target' property nor the 'href' property; that only makes sense for a page, so these input fields are not displayed for a section.

The readonly-property is a special case. Even if the parameter `$viewonly` is TRUE, the readonly-field is displayed as 'editable'. This is because this particular field is used to toggle the viewonly mode: if a node is readonly, it cannot be edited, except the removal of the readonly attribute.

array function PageManager::get_dialogdef_edit_node(\$node_id, \$is_page, [\$viewonly = FALSE]) [line 3010]

Function Parameters:

- *int* **\$node_id** the node that is to be edited
- *bool* **\$is_page** TRUE if the dialog is for a page, FALSE is for a section
- *bool* **\$viewonly** if TRUE, all fields are 'dimmed' (uneditable) and there is no [Save] button

construct a dialog definition for editing basic properties of an existing node (page or section)

the dialog for pages and sections is different in just a single field: the page has an extra module field.

Note that we return a keyed array using the name of the dialog field as a key. This makes it easier to reference an incoming field in the save routine.

array function PageManager::get_dialogdef_force_unlock() [*line 4157*]

construct a dialog definition to show [OK] and [Cancel]

this constructs the dialogdef to show two buttons AND protect against CSRF.

void function PageManager::get_dialog_data_node(&\$dialogdef, \$node_id) [*line 3221*]

Function Parameters:

- *array* **&\$dialogdef** the dialog definition
- *int* **\$node_id** the node that needs to be edited

fill the node dialog with data from the database

this fills a node dialog with data from the database. The routine takes care of some data conversions, e.g. manipulating a boolean TRUE/FALSE so it fits in a checkbox type of widget, etc.

Note that the data is NOT specifically validated. This means that a dialog *_could_* contain invalid values even when the user doesn't change anything. Or, to put it a different way: if the database contains garbage, the garbage is simply presented to the user. If the user subsequently tries to save the "garbage" the validation will catch her.

This routine is able to fill the values for both the 'basic' and the 'advanced' dialogs.

void function PageManager::get_icon_delete(\$node_id) [*line 2636*]

Function Parameters:

- *int* **\$node_id** the node to delete

construct a clickable icon to delete this node (and underlying nodes too)

- **TODO** should we display trash can icons for sections with non-empty subsections? there really is no point, because we eventually will not accept deletion of sections with grandchildren in `task_node_delete`. Hmmmmm..... For now I just added the condition that access is denied when a section has grandchildren. Need to refine this, later. Also, how about readonly nodes? Surely those cannot be deleted... should it not show in the icon?
- **Uses** `$WAS_SCRIPT_NAME`
- **Uses** `$CFG`

void function PageManager::get_icon_edit(\$node_id) [*line 2664*]

Function Parameters:

- *int* **\$node_id** the node to edit

construct a clickable icon to edit this node

- **Uses** `$WAS_SCRIPT_NAME`
- **Uses** `$CFG`

void function PageManager::get_icon_home(\$node_id) [*line 2593*]

Function Parameters:

- *int* **\$node_id** the node of interest

construct a clickable icon to set the home page/section on this tree level

this constructs a clickable icon to change the default node on this level. it requires PERMISSION_NODE_EDIT_PAGE or PERMISSION_NODE_EDIT_SECTION for both the target default node AND the current default node (if any)

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

void function PageManager::get_icon_invisibility(\$node_id) [line 2700]

Function Parameters:

- *int* **\$node_id** the node to edit

construct a clickable icon to edit the advanced properties of this node

This icon has another purpose besides creating a link to the advanced properties: it also indicates wheter a node is 'invisible' or not. In this context 'invisible' means either

- the node is under embargo until some time in the future, OR
- the node is already expired some time in the past, OR
- the node is hidden (ie. page is not visible in navigation but otherwise available).

Depending on the visibility a different icon is displayed.

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

void function PageManager::get_icon_page_preview(\$node_id) [line 2761]

Function Parameters:

- *int* **\$node_id** the node to preview

construct a clickable icon to preview this node

this constructs an icon to preview the page. the user should have edit permissions OR

edit content permissions, because you can see the page when you can edit it, so there's no point in preventing the preview in that case. See [task_page_preview\(\)](#) for more information.

The preview is displayed in a separate window, either generated via a small routing in javascript or (if javascript disabled) via a target="_blank".

- **TODO** if this is a public area, the user can see every page, except the expired/embargo'ed ones should we take that into account too? I'd say that is way over the top. How about pages in an intranet where the user has view privilege? Complicated. KISS: only show preview to those that can edit or edit content.
- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

void function PageManager::get_icon_section(\$node_id) [line 2804]

Function Parameters:

- *int* **\$node_id** the section to open/close

construct a clickable icon to open/close this node

This is a toggle: if the node is closed the closed icon is shown, but the action in the A-tag is to open the icon (and vice versa).

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

void function PageManager::get_link_node_edit(\$node_id, &\$modules) [line 2842]

Function Parameters:

- *int* **\$node_id** the node for which to make the link
- *array* **&\$modules** translates module_ids to readable text

construct a clickable link to edit this page OR open/close this section

this generates an A tag which leads to editing the content of the node (node == page) OR opens/closes the section (node == section). Additional information displayed via the title attribute includes the node_id.

Note that a user is always allowed to open/close a section so in case of a section \$user_has_permission is always TRUE.

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

array function PageManager::get_node_id_and_ancestors(\$node_id) [*line 3840*]

Function Parameters:

- *int* **\$node_id** start at the youngest in the family

get an array with all ids of ancestors of node_id and node_id itself

note that the order of nodes is from top to bottom

array function PageManager::get_options_area(\$node_id, \$is_page) [*line 3445*]

Function Parameters:

- *int* **\$node_id** the node for which we are building this picklist
- *bool* **\$is_page** TRUE if this concerns a page, FALSE for a section

generate a list of areas for use in a dropdown list (for moving a node to another area)

this creates an array containing a list of areas to which the user is allowed to move a node. Permissions for moving a node is a combination of permissions for deleting a node from the current area, and adding a node to the target area. The current area \$this->area_id is always in the list, because even if the user isn't allowed to move a node to somewhere else, she is at least allowed to leave the node in the area it currently is in. Therefore the option for the current area **MUST** be possible.

We sepcifically check for these permissions: PERMISSION_AREA_ADD_PAGE or PERMISSION_AREA_ADD_SECTION and not PERMISSION_NODE_ADD_PAGE or PERMISSION_NODE_ADD_SECTION because the target of the move is always the top level, and not some (sub)section.

array function PageManager::get_options_modules() [*line 3480*]

fetch a list of available modules for inclusion on a page

this retrieves a list of modules that can be used as a list of options in a listbox or radiobuttons. Only the active modules are considered. The names of the modules that are displayed in the list are translated (retrieved from the modules language files). The list is ordered by that translated module name.

array function PageManager::get_options_parents(\$is_page, [\$forbidden_id = NULL]) [*line 3296*]

Function Parameters:

- *bool* **\$is_page** if TRUE check page permissions, else check section permissions
- *mixed* **\$forbidden_id** identifies the subtree to EXclude from the results or NULL for all sections

construct an options list of possible parent sections

this constructs an array suitable for a radio field or a listbox. If the user has the privilege, an option 'add to toplevel' is added too.

If \$forbidden_id is not NULL, it identifies the subtree that should be excluded from the result. If it were not excluded, the user might choose a child section as the parent for a section, which would introduce endless loops or circular references. Excluding the 'own' subtree prevents that.

Note that the list is constructed using recursion: the actual work is is done in the routine [get_options_parents_walk\(\)](#).

Also note that if \$forbidden_id is not NULL, we interpret this as a request to generate a picklist of parents for that node. We make sure that we always add the current parent node to the list. This way the only option for a parent might be to keep the current one, which obviously should be one of the options.

- Uses [PageManager::get_options_parents_walk\(\)](#)

void function PageManager::get_options_parents_walk(&\$options, \$is_page, \$node_id, \$forbidden_id) [line 3342]

Function Parameters:

- *array &\$options* resulting array, output of this routine
- *bool \$is_page* distinction between page (TRUE) or section (FALSE)
- *int \$node_id* the subtree where we should start
- *int|null \$forbidden_id* if not NULL the subtree to skip

workhorse for construction an options list of possible parent sections

This routine is called recursively in order to construct a list of possible parent sections in the same order as the main tree display (see [show_tree\(\)](#)), but excluding the subtree starting at \$forbidden_id.

The list of parents is collected in \$options. This variable is passed by reference to save memory and also to keep the parents in the correct order.

Note that the options in the output array all have a parameter 'class' which can be used to detect how deep the nesting is. This can be visualised via wellchosen CSS-parameters, eg.

```
option.level0 { margin-left: 0px; }  
option.level1 { margin-left: 20px; }  
option.level2 { margin-left: 40px; }  
...
```

which provides the illusion of a tree-like structure, even in a listbox.

The current parent of node \$forbidden_id is always included in the list of allowable parents because a node should be able to keep the current parent, always.

- **Usedby** [PageManager::get_options_parents_walk\(\)](#)
- **Usedby** [PageManager::get_options_parents\(\)](#)
- **Uses** [PageManager::get_options_parents_walk\(\)](#)

array function PageManager::get_options_sort_order(\$node_id) [line 3392]

Function Parameters:

- *int \$node_id* the node for which the list of siblings must be constructed

generate a list of siblings in a particular (sub)section used to select/change sort order via a list box

this constructs an (ordered) list of siblings of \$node_id, but excluding \$node_id itself. Also, an option 'sort at the top of the list' is included. This allows for selecting a sibling AFTER which \$node_id should appear in the section. The special value for 'before all others' or 'at the top of the list' is 0, because that value cannot be used by a real node.

- **Uses \$CFG**

bool function PageManager::lock_records_subtree(\$node_id, \$new_area_id) [*line 3536*]

Function Parameters:

- *int* **\$node_id** the node which we are going to move to \$new_area_id
- *int* **\$new_area_id** the area to which we want to move the subtree \$node_id

attempt to lock all node records in a subtree

this recursively walks the subtree starting at \$node_id and attempts to

- lock every node in the subtree, AND
- write the new area_id in an auxiliary field of every node in the subtree

With at least two trips to the database (at least one for the lock and another one for writing the auxiliary field) per node, this is an expensive routine. Maybe it is possible to combine locking and writing the auxiliary field. However, in order to keep things readable I decided against that.

This routine returns FALSE if any of the nodes in the subtree could NOT be locked. If each and every node in the subtree is successfully locked, TRUE is returned.

Note that all these locks are reset/released the moment the actual move is done, by resetting both the locked_by field and the area_id field. That may hurt readability too, but less than combining lock + setting auxiliary field. See [save_node_new_area_mass_move\(\)](#) for more information.

- **Usedby** [PageManager::lock_records_subtree\(\)](#)

- **Usedby** [PageManager::save_node_new_area_mass_move\(\)](#)
- **Uses** [PageManager::lock_records_subtree\(\)](#)

string function PageManager::message_from_lockinfo(\$lockinfo, \$node_id, \$is_page) [*line 3871*]

Function Parameters:

- *array* **\$lockinfo** contains information about another user that has obtained a record lock
- *int* **\$node_id** the node that is locked
- **\$is_page**

construct a readable message from the lockinfo array

if an attempt to lock a record fails (see [lock_record_node\(\)](#)), the array \$lockinfo is filled with information about the user that has locked the record. The following information is available:

- 'user_id': the numerical user_id of the user holding the lock
- 'username': the userid of that user
- 'full_name': the full name of that user
- 'user_information': the IP-address from where that user is calling
- 'ctime': the date/time that user logged in (c=create)
- 'atime': the date/time that user last accessed the system (a=access)
- 'ltime': the date/time that user actually locked the record (l=lock)

This routine tries to construct a more or less readable message which informs this user here about that other user holding the lock.

bool function PageManager::module_connect(\$area_id, \$node_id, \$module_id) [*line 4231*]

Function Parameters:

- *int* **\$area_id** the area where \$node_id resides
- *int* **\$node_id** the node to which the module will be connected
- *int* **\$module_id** the module that will be connected to node \$node_id

inform module \$module_id that from now on it will be linked to page \$node_id

this routine tells module \$module_id that from now on it is associated with node \$node_id in area \$area_id.

This is done by a. loading the module's administrative interface (the admin-script file), and b. calling the function `<modulename>_connect()`

If something goes wrong (e.g. no module found, non-existing admin-script, undefined function `<modulename>_connect()`) `FALSE` is returned, otherwise the return value of function `<modulename>_connect()` is returned.

- **TODO** should we pass the `area_id` at all? What happens when a node is moved to another area without informing the module? Questions, questions, questions...

bool function PageManager::module_disconnect(\$area_id, \$node_id, \$module_id) [*line 4192*]

Function Parameters:

- *int* **\$area_id** the area where \$node_id resides
- *int* **\$node_id** the node from which the module is disconnected
- *int* **\$module_id** the module that will be disconnected from node \$node_id

inform module \$module_id that it is no longer linked to page \$node_id

this routine tells module \$module_id that it is no longer associated with node \$node_id in area \$area_id.

This is done by a. loading the module's administrative interface (the admin-script file), and b. calling the function `<modulename>_disconnect()`

If something goes wrong (e.g. no permissions, no module found, non-existing admin-script, undefined function `<modulename>_disconnect()`) `FALSE` is returned, otherwise the return value of function `<modulename>_disconnect()` is returned.

- **TODO** should we pass the `area_id` at all? What happens when a node is moved to another area without informing the module? Questions, questions, questions...

bool|array function PageManager::module_load_admin(\$module_id) [*line 4378*]

Function Parameters:

- *int* **\$module_id** indicates which module to load

load the admin interface of a module in core

this includes the 'admin'-part of a module via 'require_once()'. This routine first figures out if the admin-script file actually exists before the file is included. Also, we look at a very specific location, namely:
 /program/modules/<modulename>/<module_admin_script> where
 <modulename> is retrieved from the modules table in the database.

Note that if modulename would somehow be something like
 ".././.././.././etc/passwd\x00", we could be in trouble...

- **TODO** should we sanitise the modulename here? It is not user input, but it comes from the modules table in the database. However, if a module name would contain sequences of
 "../" we might be in trouble

bool function PageManager::module_save(\$node_id, \$module_id, \$viewonly, &\$amp;edit_again, \$option) [line 4330]

Function Parameters:

- *int* **\$node_id** the node to which the module is connected
- *int* **\$module_id** the module that is connected to node \$node_id
- *bool* **\$viewonly** if TRUE, editing and thus saving is not allowed
- *bool* **&\$edit_again** returns TRUE if more editing is required, FALSE otherwise
- *string* **\$option** the selected submenu option or NULL if 'Content' option was selected

(maybe) save the modified content of module \$module_id connected to page \$node_id

this saves the module data belonging to node \$node_id.

If something goes wrong (e.g. no module found, non-existing admin-script, undefined function <modulename>_save()) FALSE is returned, otherwise the return value of function <modulename>_save() is returned.

bool function PageManager::module_show_edit(\$node_id, \$module_id, \$viewonly, \$edit_again, \$option) [*line 4277*]

Function Parameters:

- *int* **\$node_id** the node to which the module is connected
- *int* **\$module_id** the module that is connected to node \$node_id
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE, start with data from \$_POST, otherwise read from database
- *string* **\$option** the selected submenu option or NULL if 'Content' option was selected

show a dialog for editing the content of module \$module_id linked to page \$node_id

this loads the code for module \$module_id and calls the appropriate routine for displaying a dialog

The parameter \$viewonly can be used to indicate readonly access to the content. It is upto the called function to adhere to this flag, e.g. by just showing the content instead of letting the user modify it.

If the flag \$edit_again is TRUE, this is not the first call to this routine, i.e. we have been here before but probably something went wrong when saving the data (e.g. an invalid date like 2008-02-31 was entered). This makes it possible to re-edit the content without starting from scratch again. If the flag is FALSE, the called routine is supposed to start with the data as it is currently stored in the database. Otherwise the current data is POST'ed by the user.

If something goes wrong (e.g. no module found, non-existing admin-script, undefined function <modulename>_show_edit()) FALSE is returned, otherwise the return value of function <modulename>_show_edit() is returned.

- **Usedby** [PageManager::task_node_edit_content\(\)](#)

string function PageManager::node_full_name(\$node_id) [*line 3805*]

Function Parameters:

- *int* **\$node_id** get the full name of this node

shorthand for constructing a readable page/section name with id, name and title

bool function PageManager::node_has_grandchildren(\$node_id) [*line 3817*]

Function Parameters:

- *int* **\$node_id** the section to check for grandchildren

shorthand to determine whether the number of levels below section \$node_id is greater than one

bool function PageManager::permission_add_any_node(\$is_page) [*line 3598*]

Function Parameters:

- *bool* **\$is_page** selects either page or section permissions

does the user have the privilege to add a node, any node to an area?

this routine returns TRUE if the current user has permission to add at least one node to the current area. This information is used to show or suppress the 'add a page' and 'add a section' links.

Note that pages and sections are treated separately; if a user is allowed to add a page it doesn't necessarily mean that she is allowed to add a section too.

Strategy: we first check the area-level (and implicit site-level) permissions to add a node, anywhere in an area including at the toplevel. If that doesn't work, we check for permissions to add a node to an existing section at the node level (and implicit at the area and site level too). If that doesn't work, we return FALSE.

Note that it is enough to stop the search at the first hit: we need only 1 hit for 'any', not all of them.

- **Uses \$USER**

bool function PageManager::permission_add_node(\$section_id, \$is_page) [*line 3630*]

Function Parameters:

- *int* **\$section_id** is the section to examine

- *bool* **\$is_page** selects either page or section permissions

does the user have the privilege to add a node to an area or a section?

this checks for permission to add a page or a section to the area at the toplevel or to the section `$section_id`. If access is denied initially, the ancestors are tested for the requested permission. I dubbed this cascading permissions (if a section allows for adding a page, any subsections inherit that permission). This routine is protected from endless loops by iterating at most `MAXIMUM_ITERATIONS` levels.

- **Uses** `$USER`

bool function PageManager::permission_delete_node(`$node_id`, `$is_page`) [*line 3751*]

Function Parameters:

- *int* **\$node_id** is the node to delete
- **\$is_page**

does the user have the privilege to delete a node from the area?

Note: Top-level nodes are a special case: a user needs to have area-wide permission to drop such a node. However, child-nodes can be deleted based on the (cascading) permissions of a top-level node.

- **Uses** `$USER`

bool function PageManager::permission_edit_node(`$node_id`, `$is_page`, [`$check_content = FALSE`]) [*line 3680*]

Function Parameters:

- *int* **\$node_id** is the node to examine
- *bool* **\$is_page** TRUE means we look at page permissions, not section

- *bool* **\$check_content** TRUE means check edit content, else edit plain

does the user have the privilege to edit node properties?

this checks the edit permissions for the specified node and the node's ancestors. If none are found initially, we check out the add permissions at the parent and parent's ancestors.

Note that a node can also have the readonly attribute set. This is more or less a tool to prevent accidental changes to a node's properties: a user can easily reset the readonly flag and change the node anyway. However, it requires two steps and hence at least *_some_* thinking. Bottom line: we only look at the 'real' permissions here, and not the readonly flag. (Even better: edit privilege is required to reset the readonly flag so using that flag as extra permission would yield pages completely uneditable).

This routine is also used to check for content edit permissions. This is only possible for pages (not sections). By default this routine checks the regular permissions (edit properties/edit advanced properties).

- **Uses \$USER**

bool function PageManager::permission_edit_node_content(\$node_id) [*line 3712*]

Function Parameters:

- *int* **\$node_id** is the node to examine

does the user have the privilege to edit node content?

this is a wrapper around routine [permission_edit_node\(\)](#). We force *is_page* and *check_content* to TRUE.

bool function PageManager::permission_set_default(\$node_id) [*line 3727*]

Function Parameters:

- *bool* **\$node_id** is the tentative new default

does the user have the privilege to make node \$node_id the default?

if a user has edit permission for the new default node and also in the existing default node (if any), the user is allowed to set the default to node \$node_id. Note that once again we use cascading permissions. (See also [permission_edit_node\(\)](#)).

void function PageManager::save_node(\$node_id, \$done) [line 2169]

Function Parameters:

- *int \$node_id* the node we have to change
- *bool \$done* TRUE means return to treeview, FALSE is stay in edit mode (and keep lock!)

workhorse routing for saving modified node data to the database

this is the 'meat' in saving the modified node data. There are a lot of complicated things we need to take care of, including dealing with the readonly property (if a node is currently readonly, nothing should be changed whatsoever, except removing the readonly attribute) and with moving a non-empty section to another area. Especially the latter is not trivial to do, therefore it is being done in a separate routine (see [save_node_new_area_mass_move\(\)](#)).

Note that we need to return the user to the edit dialog if the data entered is somehow incorrect. If everything is OK, we simply display the treeview and the area menu, as usual.

Another complication is dealing with a changed module. If the user decides to change the module, we need to inform the old module that it is no longer connected to this page and is effectively 'deleted'. Subsequently we have to tell the new module that it is in fact now added to this node. It is up to the module's code to deal with these removals and additions (for some modules it could boil down to a no-op).

Finally there is a complication with parent nodes and sort order. The sort order is specified by the user via selecting the node AFTER which this node should be positioned. However, this list of nodes is created based on the OLD parent of the node. If the node is moved to elsewhere in the tree, sorting after a node in another branch no longer makes sense. Therefore, if both the parent and the sort order are changed, the parent prevails (and the sort order information is discarded).

Update 2014-04-29: we now distinguish between 'Save' and 'Done': in the latter case we show the tree once the data is saved, in the former we redo the edit dialog again. This allows for frequent saving without 'losing' the page you are editing (and having to look it up again in the tree).

- **TODO** this routine could be improved by refactoring it; it is too long!

- **TODO** there is something wrong with embargo: should we check starting at parent or at node? this is not clear: it depends on basic/advanced and whether the embargo field changed. mmmm... safe choice: start at node_id for the time being

bool function PageManager::save_node_new_area_mass_move(\$node_id, \$new_area_id, \$embargo) [line 2484]

Function Parameters:

- *int* **\$node_id** the node which we are going to move to \$new_area_id
- *int* **\$new_area_id** the area to which we want to move the subtree \$node_id
- *bool* **\$embargo** if TRUE, we cannot send alerts because the original tree is under embargo

workhorse routine for moving a complete subtree to another area

this routine moves a subtree starting at section \$node_id from area \$this->area_id to area \$new_area_id. This is a complicated operation because

- the subtree may be (very) large
- nodes in the subtree may be locked by other users
- we **MUST** take an all or nothing approach because either ALL of the nodes or NONE of the nodes

in the subtree must change the area_id. If area_id's are only changed partly, we will end up

with orphan nodes because areas differ between a parent and corresponding offspring.

Here is my train of thoughts leading to my implementation.

The best solution I can think of is to:

- lock all individual nodes in the subtree, and if successful,
- update all these records with the appropriate area_id and at the same time unlocking these records by writing a NULL to the lock field.

This way we postpone the actual move to the new area until we are certain that we have all nodes involved in our hands. If we don't succeed in obtaining all the necessary locks, we have to abandon the operation, accept defeat, release all locks and return FALSE to our caller. If we do succeed, well, we indicate success by returning TRUE.

Mmmm....

Note that the user might have two browser windows open in the same session. This shouldn't happen, but there is no easy way to prevent the user to open more windows in the same session. This may lead to an undesirable result: if the user is editing another node in the same session in another window (totally unrelated to the move of the current subtree), that node might also be moved to the new area, introducing an orphan in the new area. Mmmmm. The best way to handle that problem is to use a special helper field, say auxiliary_id, in the nodes table. That field could be used as follows (pseudo-code):

set auxiliary_id of \$node_id to \$new_area_id for all descendants of section \$node_id do
obtain lock on descendant set auxiliary_id to \$new_area_id update nodes set area_id =
new_area_id, auxiliary_id = NULL, locked_by = NULL where auxiliary_id = new_area_id AND
locked_by = our_session_id Then we once again concentrate the actual work in a single
UPDATE-statement.

It is a costly operation: at least 2 trips to the database per descendant.

Mmmmm...

Perhaps we can save (a lot) of trips to the database if we build on the assumption that usually
there are more children in every section AND that usually the children are NOT locked. In
that case the pseudo-code becomes:

```
for all descendants of section $node_id do    if is_section($descendant) then        SET  
auxiliary_id = $new_area_id, locked_by = $our_session_id WHERE                    locked_by IS  
NULL AND parent_id = $descendant;    endif endfor SET area_id = new_area_id, auxiliary_id  
= NULL, locked_by = NULL    WHERE auxiliary_id = new_area_id AND locked_by =  
$our_session_id
```

However, we might miss a descendant or two if it happens to be locked (by us, or by another
session). That's no good.

Mmmmm...

I'm sure there's a better way, but for the time being I'll simply use brute force and my way
through the subtree. If this really becomes a huge problem, we may want to refactor this
routine.

- Uses [PageManager::lock_records_subtree\(\)](#)

bool function PageManager::section_is_open(\$section_id) [*line 3785*]

Function Parameters:

- *int* **\$section_id** the section of interest

shorthand for determining whether a section is opened or closed

void function PageManager::show_area_menu([\$current_area_id = NULL]) [*line 1532*]

Function Parameters:

- *int*/*null* **\$current_area_id** the current area

construct a clickable list of available areas for the current user

this iterates through all available areas in the areas table, and constructs a list of areas (as LI's in a UL) for which the current user has either administrative or view permissions. The latter shows in 'dimmed' form, because it is not allowed to view this area in pagemanager, but the area does exist and is available to the user (as a visitor rather than an administrator) so it should not be suppressed. If a user has neither view or admin permission, the area is suppressed. Note that every user has at least view permissions for a public area.

The current area is determined by parameter `$current_area_id`. This area gets the attribute `'class="current"'` which makes it possible to emphasise the current working area in the menu (via CSS).

void function PageManager::show_dialog_delete_node_confirm(\$node_id, \$dialogdef) [*line 1988*]

Function Parameters:

- *int* **\$node_id** the page or the section to delete
- *array* **\$dialogdef** defines the confirmation dialog

display a list of 1 or more nodes to delete and ask user for confirmation of delete

this displays a confirmation question with a list of nodes that will be deleted. This list is either a single page or a single (empty) section OR a section with children (but not grandchildren). See function [task_node_delete\(\)](#) for more on this design decision. If the user presses Delete button, the nodes will be deleted, if the user presses Cancel then nothing is deleted.

output function PageManager::show_dialog_force_unlock(\$node_id, \$lockinfo) [*line 4117*]

Function Parameters:

- *int* **\$node_id** the locked node we want to grab
- *array* **\$lockinfo** has information about the current locker

show a dialog to the user offering to forcefully unlock a node

`void function PageManager::show_edit_menu($node_id, [$is_page = FALSE], [$current_option = NULL],
[$current_submenu = NULL]) [line 1597]`

Function Parameters:

- *int* **\$node_id** the current node (the node being edited)
- *bool* **\$is_page** if TRUE display the link to edit content too (this is for pages only)
- *string* **\$current_option** the currently selected edit mode (basic, advanced or content)
- *string* **\$current_submenu** currently selected content submenu option or NULL for just content

construct a clickable list of edit variants (basic, advanced and maybe content)

this constructs a menu from where the user can navigate to edit basic properties of a node, advanced properties or even the content (for pages).

Update 2014-05-05. Depending on the module there may be additional submenu items to show beneath the Content menu entry. These are defined via the record in the modules table. Note that this routine expects translations of the submenu items in the domain of the module.

These translation keys are:

`<option>_title`
`<option>_anchor`

and they should be defined in the module's translations.

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

`void function PageManager::show_tree() [line 1705]`

create a tree-like list of nodes in the content area of \$this->output

this constructs a tree-like view of the current area, with

- a title
- 0, 1 or 2 links to add a node
- 0, 1 or 2 links to select a different tree view
- all nodes that are currently show-able (depending on tree view mode)

If the tree is empty, only the links to add a node are displayed (if the user has permission to add). The individual nodes are displayed using recursion with [show tree walk\(\)](#).

Note that the tree is constructed via nested UL's with LI's, all in name of 'graceful degradation': this interface still works if this program has no stylesheet whatsoever).

- Uses [PageManager::show_tree_walk\(\)](#)
- Uses \$WAS_SCRIPT_NAME
- Uses \$CFG

void function PageManager::show_treeview_buttons() [line 1904]

show one or two clickable links to change the view of the tree

There are three different tree views:

- minimal: all sections are closed, only the top level nodes are shown
- custom: 1 or more sections are closed and 1 or more sections are opened
- maximal: all sections are opened, all nodes are shown

There is a fourth option:

- none: there are no sections at all

The view can be set to either TREE_VIEW_MINIMAL, TREE_VIEW_CUSTOM or TREE_VIEW_MAXIMAL. The current setting is remembered in session variable 'tree_mode'. A list of customised nodes is kept in session variable expanded_nodes[], an array keyed with the node number and a value of either TRUE (section is 'open') or FALSE (section is 'closed'). An empty array implies all nodes are closed (ie. default value is FALSE).

In some cases TREE_VIEW_CUSTOM is equivalent to one of the other two, e.g. when the user closes the last section, the effect looks exactly like TREE_VIEW_MINIMAL. If the user manually opens all sections, the effect is the same as TREE_VIEW_MAXIMAL.

In this routine we want to show 0, 1 or 2 buttons that allow the user to switch to another viewmode, but only if the new mode(s) are different from the current one.

The equivalency between modes can be determined by counting the number of open and closed sections. Here is a truth table.

N	open	closed	description
0	0	0	no sections at all, show 0 buttons (all modes are equivalent)
1	0	>=1	all sections are closed, 'custom' is equivalent with 'minimal'
2	>=1	0	all sections are opened, 'custom' is equivalent with 'maximal'
3	>=1	>=1	some open, some closed, 'custom' is distinct from the other two modes

Case N=0 In this case there are no sections at all, so there is no point to show any button at all because all views are equivalent: all available pages (if any) live at the top level and they are always visible.

Case N=1 In this case 'minimal' and 'custom' are equivalent. That means that if the current view is either 'minimal' or 'custom', the only viable option would be to set the view to 'maximal'. If the current mode is 'maximal', the only viable option is 'minimal'. Only 1 toggle-like button needs to be displayed.

Case N=2 In this case 'custom' and 'maximal' are equivalent. That means that if the current view is either 'custom' or 'maximal', the only viable option would be to set the view to 'minimal'. If the current mode is 'minimal', the only viable option is 'maximal'. Only 1 toggle-like button needs to be displayed.

Case N=3 In this case 'custom' is a distinct mode somewhere between 'minimal' and 'maximal'. This means that there are always two other options to choose from: if current mode is 'minimal' the choices are 'custom' and 'maximal', if current mode is 'custom' the choices are 'maximal' and 'minimal', if current mode is 'maximal' the choices are 'minimal' and 'custom'. This means that two buttons need to be displayed.

Strategy: First we step through the tree and we count the 'open' and 'closed' sections. After that we determine whether N is 0,1,2 or 3 (see truthtable). After that we calculate which of the three buttons need to be displayed, depending on the current mode (obtained via the session variable 'tree_mode'). Subsequently the buttons are output to the 'content' area via `$this->output`.

- **Uses** `$WAS_SCRIPT_NAME`

void function `PageManager::show_tree_walk($node_id, &$modules, [$m = ""])` [*line 1789*]

Function Parameters:

- *int* **`$node_id`** the first node of this tree level to show
- *array* **`&$modules`** translates module_ids to readable text
- *string* **`$m`** left margin for increased readability

display the specified node, optionally all subtrees, and subsequently all siblings

this routine displays the specified node, including clickable icons for setting the default, editing the node etc. from the current tree. After that, any subtrees of this node are displayed using recursion (but only if the section is 'opened'). This continues for all siblings of the specified node until there are no more (indicated by a sibling_id equal to zero).

- **Usedby** [PageManager::show_tree_walk\(\)](#)

- **Usedby** [PageManager::show_tree\(\)](#)
- **Uses** [PageManager::show_tree_walk\(\)](#)
- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

void function PageManager::task_force_unlock() [line 1445]

forcefully obtain a lock on a node and release it immediately

this routine attempts to steal the lock on node \$node_id if this node is locked by our user_id (but in another session) and releases it immediately. The effect is that the node is forcefully unlocked. This deals with the annoying problem of a crashed browser and a node that remains locked until that dead session times out. The safety precautions are that we can only grab a session that is currently in possession of the same user_id. IOW: it is not possible to unlock another user's lock.

void function PageManager::task_node_add(\$task) [line 560]

Function Parameters:

- *string* **\$task** identifies whether a page or a section should be added

display a dialog to add a new page or section to the current area

this displays a dialog where the user can add a node to the current area. If the user has no permissions to add a node at all, the result is an error message and the tree view

The value of \$task (which can be either TASK_ADD_PAGE or TASK_ADD_SECTION) determines which dialog to show.

Both dialogs are very similar (a page can have a module, a section cannot). The actual dialog is constructed based on a dialogdef, see the function [get_dialogdef_add_node\(\)](#).

void function PageManager::task_node_delete() [line 635]

delete one or more nodes from an area after user confirmation

this deals with deleting nodes from an area. There are two stages. Stage 1 is presenting the user with a list of selected nodes and offering the user the choice to confirm the delete or cancel the operation. Stage 2 is actually deleting the selected nodes (after the user confirmed the delete in stage 1), including the disconnection of pages and modules.

An important design decision was to limit the delete process to at most 1 tree level. This means the following. If a user attempts to delete a page, it is easy: after confirmation a single

node is deleted from the database. If a user attempts to delete a section, it can be different. If the section is empty, i.e. there are no children, it is the same as deleting a page: only a single node record has to be deleted.

It becomes more dangerous if a section is filled, ie. has children. If all children are pages (or empty subsections), it is still relatively innocent because the worst case is that all pages in a section are deleted. If, however, the section contains subsections which in turn contain subsections, etc. the delete operation may become a little too powerful. If it would work that way (deleting a section implies `_all_` nodes in the subtree), it is possible to delete a complete area in only a few keystrokes, no matter how many levels.

In order to prevent this mass deletion, we decided to limit the delete operation to at most a single level. In other words: the user can delete

- a single page
- a single empty section
- a section with children but no grandchildren

If a user attempts to delete a section with children and grandchildren, an error message is displayed and nothing is deleted.

This forces the user to delete a complete tree a section at the time, hopefully preventing a 'oh no! what have I done' user experience.

We `_always_` want the user to confirm the deletion of a node, even if it is just a single page.

Note that a page that is readonly will not be deleted.

- **TODO** should we display trash can icons for sections with non-empty subsections in treeview? there really is no point, because we eventually will not accept deletion of sections with grandchildren. Hmmmmm.....

`void function PageManager::task_node_edit($task, $node_id) [line 751]`

Function Parameters:

- *string* **\$task** identifies whether the basic or advanced properties should be edited
- *int* **\$node_id** identifies the node to edit

display a dialog where the user can edit basic or advanced properties of a node

this constructs a dialog and a menu where the user can edit the properties of a node. We check the user's permissions and if that works out we try to obtain a lock on the record. If that succeeds, we show the dialog (in funnel mode). If we don't get the lock, we inform the user

about the other user who holds the lock. In case of error (e.g. no permissions or no lock) we fall back on displaying the area menu and the treeview.

Note: the lock is released once the user saves the node OR cancels the edit operation.

void function PageManager::task_node_edit_content() [line 845]

display a dialog where the user can edit the contents of a node via a module

this effectively loads the module code associated with the specified node and subsequently calls the corresponding code in the module to display an edit dialog.

Just like the other edit routine (see [task_node_edit\(\)](#)) the node is locked first. Also the user permissions are checked. If we don't get the lock, we inform the user about the other user who holds the lock. In case of error (e.g. no permissions or no lock or an error loading the module) we fall back on displaying the area menu and the treeview. In that process the lock may be released.

Update 2014-05-05. There were two changes in the module interface, see also . One was the addition of a [Done]-button and the changed meaning of the [Save] button. The other is about extra options in the edit menu. For pages it used to be Basic, Advanced or Content. We now have a way to add submenu options to the Content menu option. This is done via a comma delimited list of possible options for a module. This list is stored in the new options field in the modules table. If the list is empty there are no additional submenu options to show (in [show_edit_menu\(\)](#) or to use. If there is at least one, the options are added to the edit menu and also conveyed via the command line (in PARAM_SUBMENU_OPTION). So, the effect is that there are now multiple admin edit screens per module, each with their own PARAM_SUBMENU_OPTION. It is up to the module to use that information and display the correct edit screen (and perform the correct save routine).

- Uses [PageManager::module_show_edit\(\)](#)
- Uses \$USER

void function PageManager::task_page_preview() [line 979]

preview a page that is maybe still under embargo/already expired

if the user has permissions to preview the specified page, she is redirected to the regular site with a special one-time permission to view a page, even if that page is under embargo or already expired (which normally would prevent any user from viewing that page).

There are several ways to implement such a one-off permit, e.g. by setting a quasi-random string in the session and specifying that string as a parameter to index.php. If (in index.php) the string provided matches this string in the session, the user is granted access. However,

this leaves room for the user to manually change the node id to `_any_` number, even a node that that user is not supposed to see.

Another solution might have been to simply include `index.php`. I decided against that; I don't want to have to deal with a mix of `admin.php` and `index.php`-code in the same run.

I took a slightly different approach, as follows. First I generate a quasi-random string of `N` (`N=32`) characters. (The length of 32 is an arbitrary choice.) This string is stored in the session variable. Then I store the requested node in the session variable, too. After that I calculate the md5sum of the combination of the random string and the node id. This yields a hash. This hash is passed on to `index.php` as the sole parameter.

Note that the quasi-random key never leaves the server: it is only stored in the session variables. Also, the node id is not one of the parameters of `index.php`, this too is only stored in the session variables.

Once `index.php` is processed, the specified md5sum is retrieved and a check is performed on the node id and the quasi-random string in the session variables in order to see if the hashes match. If this is the case, `index.php` can proceed to show the page preview. Note that there is no way for the user to manipulate the node id, because that number never travels to the user's browser in plain text.

Making a bookmark for the preview will use the hash, but the hash depends on a quasi-random string stored in the session. It means that when the session is terminated, the bookmarked page will no longer be visible, which is good. Also, whenever another page preview is requested, a new quasi-random string is generated, which also invalidates the bookmarked page.

The only thing that CAN happen is that the user saves the preview in a place where it can be seen by others. Also, the page will probably be cached in the user's browser.

With respect to permissions: I consider the preview privilege equivalent with edit permission: if the user is able to edit the node she can see the content of the node anyway. However, maybe we should look at different permissions. Put it on the todo-list.

- **TODO** the check on permissions can be improved (is `PERMISSION_XXXX_EDIT_NODE` enough?)
- **TODO** there is an issue with redirecting to another site: officially the url should be fully qualified (ie. `$CFG->www`). I use the shorthand, possibly without scheme and hostname (`$CFG->www_short`). This might pose a problem with picky browsers. See [calculate_uri_shortcuts](#) for more information.
- **Uses** `$CFG`

```
void function PageManager::task_save_content() [line 1323]
void function PageManager::task_save_newnode($task) [line 1082]
```

Function Parameters:

- *string* **\$task** distinguishes between saving a page or a section

save a newly added node to the database

this validate and save the (minimal) data for a new node (section or page). First we check which button press brought us here; Cancel means we're done, else we need to validate the user input. This is done by setting up the same dialog structure as we did when presenting the user with a dialog in the first place. This ensures that WE determine which fields we need to look for in the `_POST` data. (If we simply were to look for fieldnames in the `_POST` array, we might be tricked in accepting random fieldnames. By starting from the dialog structure we make sure that we only look at fields that are part of the dialog; any other fields are ignored, minimising the risks of the user trying to trick us.)

The dialog structure is filled with the data POST'ed by the user and subsequently the data is validated against the rules in the dialog structure (eg. min length, min/max numerical values, etc). If one or more fields fail the tests, we redo the dialog, using the data from `_POST` as a new starting point. This makes that the user doesn't lose all other field contents if she makes a minor mistake in entering data for one field.

If all data from the dialog appears to be valid, it is copied to an array that will be used to actually insert a new record into the nodes table. This array also holds various other fields (not part of the dialog) with sensible default values. Interesting 'special' fields are 'sort_order' and 'is_hidden' and 'embargo'.

'sort_order' is calculated automatically from other sort orders in the same parent section. There are two ways to do it: always add a node at the end or the exact opposite: always add a node at the beginning. The jury is still out on which of the two is the best choice (see comments in the code below).

'is_hidden' and 'embargo' are calculated from the dialog field 'node_visibility'. The latter gives the user three options: 'visible', 'hidden' and 'embargo'. This translates to the following values for 'is_hidden' and 'embargo' (note that `$now` is the current time in the form 'yyyy-mm-dd hh:mm:ss'):

visible: `is_hidden = FALSE, 'embargo' = $now`
hidden: `is_hidden = TRUE, 'embargo' = $now`
embargo: `is_hidden = TRUE, 'embargo' = '9999-12-31 23:59:59'`

This makes sure that IF the user wants to create a 'secret' node, ie. under embargo until some time in the future, the new node is never visible until the user edits the node to make it visible. However, there is no need to manually add a date/time: we simply plug in the maximum value for a date/time, which effectively means 'forever'.

Finally, if the new node is saved, a message about this event is recorded in the logfile (even for new nodes under embargo). Also, if the node is NOT under embargo, an alert message is queued. Note that we do NOT send alerts on a page that is created under embargo. (There is a slight problem with this: once a user edits the node and sets the embargo to a more realistic value, e.g. next week, there is no practical way to inform the 'alert-watchers' about that fact: we cannot send an alert at the time that the embargo date is changed to 'next week' because the node is still under embargo. We don't have a handy opportunity to send alerts because the embargo date will eventually come around and the node will become visible automatically, without anyone being alerted to the fact. Mmmm....

- **TODO** about 'sort_order': do we insert nodes at the end or the beginning of a parent section?
- **TODO** how do we alert users that an embargo date has come around? Do we schedule alerts via cron?

void function PageManager::task_save_node() [line 1211]

void function PageManager::task_set_default() [line 465]

make the selected node the default for this level

this sets a default node. First we make sure we have a valid environment and a node that belongs to the current area Then we check permissions and if the user is allowed to

- set the default bit on the target node, AND
- reset the default bit on the current default node
we actually
- reset the default bit from the current default (if there is one), AND
- set the default bit for the selected node.

Note: if the user sets the default node on the current default node, the default is reset and subsequently set again (two trips to the database), This also updates the mtime of the record.

- **Uses \$USER**

void function PageManager::task_subtree_collapse() [line 413]

close the selected section and perhaps change the view mode

this closes the selected node, i.e. fold in the subtree starting at the selected node. This should only happen when the view mode is either maximal (all sections closed) or custom (some sections opened and some sections closed). It should never happen when mode is minimal.

The status of a node (opened or closed) is remembered in session variable 'expanded_nodes': an array keyed with node_id. If the corresponding value is TRUE, the section is considered open, all other values (FALSE or element is non-existing) equate to closed. See also [task_subtree_expand\(\)](#).

If the current mode is 'maximal', all sections are showed 'open'. When one of the sections is closed (via this routine), we change the mode to 'custom'. However, because the previous state was 'all sections are opened', we need to remember all the sections in the session variable 'expanded_nodes' and set them all to TRUE except the section that needs to be closed. We do this by constructing the complete tree of the area and adding an entry for every section and setting the value to TRUE, except the node that needs to be closed.

- **Uses \$USER**

void function PageManager::task_subtree_expand() [line 362]

open the selected section and perhaps change the view mode

this opens the selected node, i.e. unfold 1 level of the subtree starting at the selected node. This should only happen when the view mode is either minimal (all sections closed) or custom (some sections opened and some sections closed). It should never happen when mode is maximal.

The status of a node (opened or closed) is remembered in session variable 'expanded_nodes': an array keyed with node_id. If the corresponding value is TRUE, the section is considered open, all other values (FALSE or element is non-existing) equate to closed. See also [task_subtree_collapse\(\)](#).

- **Uses \$USER**

void function PageManager::task_treeview() [line 292]

maybe change the current area and then show the tree and the menu for the current area

this routine switches to a new area if one is specified and subsequently displays the tree of the new area or the existing current area.

void function PageManager::task_treeview_set() [line 328]

this sets the tree view to the specified mode

this is a simple routine to set the current view to one of the three possible views. The problem that sometimes 'custom' yields a view identical with 'maximal' or 'minimal' is dealt with when constructing the links to this routine `task_treeview_set()`. See [show treeview buttons\(\)](#) for more information.

Class Theme

[line 34]

Methods to access properties of a theme

- **Package** wascore

Theme::\$area_id

int = NULL [line 42]

- **Var** \$area_id the area to display

Theme::\$area_record

array = NULL [line 45]

- **Var** a copy of the area record from the areas table

Theme::\$breadcrumb_addendum

array = array() [line 93]

- **Var** \$breadcrumb_addendum holds additional anchors that can be set by the page's module

Theme::\$breadcrumb_separator

string = - [line 108]

- **Var** \$breadcrumb_separator is the delimiter between breadcrumbs in the breadcrumb trail

Theme::\$config

array = [line 57]

- **Var** \$config all properties from themes_areas_properties for this combo of theme+area

Theme::\$content

array = array() [line 72]

- **Var** collection of items/lines that are part of the content area

Theme::\$domain

string = [line 96]

- **Var** \$domain the language domain containing translations, usually 't_<themename>'

Theme::\$dtd

string = <!DOCTYPE html> [line 60]

- **Var** the standard doctype (default: HTML 5)

Theme::\$html_head

array = array() [line 69]

- **Var** collection of items/lines that will be output as part of the HTML-head section

Theme::\$http_headers

array = array() [line 66]

- **Var** collection of individual http-headers that are to be sent `_before_` any HTML is sent

Theme::\$jumps

array = array() [line 99]

- **Var** \$jumps holds an `area_id => area_title` pair for every area this user can access

Theme::\$messages_bottom

array = array() [line 81]

- **Var** collection of messages to be displayed via a javascript `alert()` at END of page

Theme::\$messages_inline

array = array() [line 78]

- **Var** collection of messages to be displayed inline, contained within the HTML body

Theme::\$messages_top

array = array() [line 75]

- **Var** collection of messages to be displayed via a javascript alert() at START of page

Theme::\$node_id

int = NULL [line 48]

- **Var** \$node_id the node (page) to display

Theme::\$node_record

array = NULL [line 51]

- **Var** a convenient copy of the node record copied from the area tree

Theme::\$preview_mode

bool = FALSE [line 87]

- **Var** \$preview_mode if TRUE, we are previewing a page (from pagemanager)

Theme::\$quickbottom_separator

string = [line 105]

- **Var** \$quicktop_separator is the delimiter between quicklinks at the bottom of the page

Theme::\$quicktop_separator

string = [line 102]

- **Var** \$quicktop_separator is the delimiter between quicklinks at the top of the page

Theme::\$silent_mode

bool = FALSE [line 90]

- **Var** \$silent_mode if TRUE, all output via [send_output\(\)](#) is suppressed

Theme::\$text_only

bool = FALSE [line 84]

- **Var** this switches the navigation between image-based and text-based

Theme::\$theme_id

int = NULL [line 39]

- **Var** \$theme_id primary key of the theme

Theme::\$theme_record

array = NULL [line 36]

- **Var** a copy of the corresponding record from the themes table

Theme::\$title

string = [line 63]

- **Var** the title to display in both the title tag and in the page itself (usually areaname)

Theme::\$tree

array = FALSE [line 54]

- **Var** \$tree all nodes in area \$area_id, keyed by \$node_id (see [tree_build\(\)](#)).

Constructor *void* function Theme::Theme(\$theme_record, \$area_id, \$node_id) *[line 126]*

Function Parameters:

- *array* **\$theme_record** the record straight from the database
- *int* **\$area_id** the area of interest
- *int* **\$node_id** the node that will be displayed

construct a Theme object

this stores the information about this theme from the database. Also, we construct/read the tree of nodes for this area \$area_id. This information will be used later on when constructing the navigation. The node to display is \$node_id.

Also, we prepare a list of areas where the current user is allowed to go. This is handy when constructing a jumpmenu and doing it here saves a trip to the database later on in [get_jumpmenu\(\)](#).

void function Theme::add_content(\$content) [line 937]

Function Parameters:

- *string|array* **\$content** the line(s) of text to add

add a line or array of lines to the content part of the document

void function Theme::add_html_header(\$headerline) [line 839]

Function Parameters:

- *string* **\$headerline** headerline to add

add a header to the HTML head part of the document

void function Theme::add_http_header(\$headerline) [line 829]

Function Parameters:

- *string* **\$headerline** headerline to add

add an HTTP-header

void function Theme::add_message(\$message) [line 877]

Function Parameters:

- *string|array* **\$message** message(s) to add inline

add a message to the list of inline messages, part of the BODY of the document

void function Theme::add_meta(\$meta) [line 910]

Function Parameters:

- *array* **\$meta** an array with name-value-pairs that should be added to the HTML head part

add a line with meta-information to the HTML head part of the document

Note: this code is hardly used since most meta headers are not allowed in HTML5 anyway. Maybe we can remove this routine in a later version.

void function Theme::add_meta_http_equiv(\$meta) [line 925]

Function Parameters:

- *array* **\$meta** an array with name-value-pairs that should be added to the HTML head part

add a line with http-equiv meta-information to the HTML head part of the document

Note: this code is hardly used since most meta headers are not allowed in HTML5 anyway. Maybe we can remove this routine in a later version.

void function Theme::add_popup_bottom(\$message) [line 863]

Function Parameters:

- *string/array* **\$message** message(s) to add

add a message to the list of popup-messages at the BOTTOM of the document

void function Theme::add_popup_top(\$message) [line 849]

Function Parameters:

- *string/array* **\$message** message(s) to add

add a message to the list of popup-messages at the TOP of the document

void function Theme::add_stylesheet(\$url) [line 896]

Function Parameters:

- *string* **\$url** absolute or relative url of the stylesheet (see above)

add a link to a stylesheet to the HTML head part of the document

this adds a link to a stylesheet file to the HTML head part of the document. Note that we qualify the path to prevent problems with incorrect assumptions about relative URLs, see [was_url\(\)](#).

- Uses [was_url\(\)](#)

array function Theme::calc_breadcrumb_trail(\$node_id) [line 1007]

Function Parameters:

- *int* **\$node_id** the node for which to calculate/set the path to the root node

set breadcrumbs in tree AND construct list of clickable anchors

Note: the anchors are created with the current setting of the preview mode, so if that changes after we construct a list of anchors we're in trouble. I prefer late binding, so the real list to use should be created in the phase where the HTML-code is constructed. Mmmmm...

- **TODO** split into two separate routines, one to set the tree, another to construct the list of anchors

array function Theme::construct_tree(\$area_id) [line 985]

Function Parameters:

- *int* **\$area_id** the tree is built from nodes within this area

read all nodes from table for this area and construct a tree

this constructs the tree for this area, and makes sure that only non-hidden pages and non-empty sections are visible

void function Theme::dump_subtree(\$node_id, &\$tree, &\$tree)) [line 1103]

Function Parameters:

- *int* **\$node_id** start of the subtree
- *array* **&\$tree** pointer to a tree that was built earlier
- **&\$tree**

a helper-routine during development/debugging (currently unused)

string function Theme::get_address([\$m = ""]) [*line 819*]

Function Parameters:

- *string* **\$m** left margin for increased readability

return the reconstructed URL in a single (indented) line

actual code moved to waslib; keep this wrapper for compatibility

string function Theme::get_bazaar_style_style([\$m = ""]) [*line 1149*]

Function Parameters:

- *string* **\$m** margin for increased readability

collect bazaar style style from area and nodes

this collects the additional style per node and per area This implements the Bazaar Style Style Sheets: every node can add additional style information ad hoc.

The additional information is constructed backwards so the style cascades in the correct direction, ie. from area to top-level section, subsections and finally the page.

Note that the style information is trim()'ed and that individual lines in the style information area indented by two additional spaces for increased readability. All Bazaar Style Style is concatenated within a single style-tag.

string function Theme::get_bottomline([\$m = ""]) [*line 797*]

Function Parameters:

- *string* **\$m** left margin for increased readability

show 'powered by' and (maybe) report basic performance indicators

This calculates the execution time of the script and the number of queries. Note a special trick: we retrieve the translated string in a dummy variable before calculating the number of queries because otherwise we might miss one or more query from the language/translation subsystem.

Note: for the time being the performance report commented out (2010-12-08). Update: as from 2011-05-20 the performance report only displayed while debug is on,

string function Theme::get_content([\$m = ""]) [*line 380*]

Function Parameters:

- *string* **\$m** left margin for increased readability

get all lines in the content DIV in a single properly indented string

string function Theme::get_div_breadcrumbs([\$m = ""]) [*line 473*]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct breadcrumb trail

this constructs a breadcrumb trail with clickable links. The crumbs are separated by this->breadcrumb_separator (default '-'). Note that if there is actually a non-empty separator, we append a space for readability.

string function Theme::get_div_messages([\$m = ""], [\$div_id = 'messages']) [*line 416*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *string* **\$div_id** contains id of the generated div

get a perhaps bulleted list of messages in a DIV

This constructs an unordered list with messages, if there are any. If there is no message at all, an empty string is returned (without DIV). If there is a single message, no bullet is

added to the message. If there are two or more messages, bullets are added.

Note that this routine is an exception with respect to the DIV-tags: this helper routine DOES generate its own DIVs whenever there is at least 1 message. This means that there is no DIV at all when there are no messages.

string function Theme::get_html() [*line 251*]

construct an output page in HTML

This constructs a full HTML-page, starting at the DTD and ending with the html closing tag.

The page is constructed using nested DIVs, the layout is taken care of in a separate style sheet. All knowledge about the structure of the page is contained in this routine.

The performance of the script (# of queries, execution time) is calculated as late as possible, to catch as much as we can. Therefore the construction is done in two parts and performance is calculated last.

The contents of the various DIVs is constructed in various helper routines in order to make this routine easy to read (by humans that is). The various helper routines all are called with a string of space characters; this should improve the the readability of the page that is generated eventually.

Note that the routine \$this->get_div_messages() does in fact generate its own DIV tags. This is done in order to completely get rid of the message DIV, we do not even want to see an empty DIV if there are no messages.

The same logic applies to the breadcrumb trail.

string function Theme::get_html_head([\$m = ""]) [*line 336*]

Function Parameters:

- *string* **\$m** left margin for increased readability

get all lines in the HTML head section in a single, properly indented string

string function Theme::get_jumpmenu([\$m = ""]) [*line 750*]

Function Parameters:

- *string* **\$m** add readability to output

construct a simple jumplist to navigate to other areas

this constructs a listbox with areas to which the current user has access. The user can pick an area from the list and press the [Go] button to navigate to that area. Only the active areas are displayed. Private areas are only displayed when the user actually has access to those areas.

This routine always shows the Submit-button even when JavaScript is turned 'off'. If it is 'on', a tiny snippet auto-submits the form whenever the user selects another area; no need press any button anymore. However, pressing the Go button is necessary when Javascript is 'off'. Rationale: the user will find out soon enough that pressing the button is superfluous, and as a benefit we keep the same look and feel no matter the state of Javascript.

We rely on the constructor to provide us with an array of area_id=>area_title pairs in the \$this->jumps array.

The special preview-mode is implemented by adding the necessary hash in the preview parameter via a hidden field. This will ultimately lead to ourselves, with the preview code so we can never leave for another area in preview mode.

- Uses [dialog_get_widget\(\)](#)

string function Theme::get_lines(\$lines, [\$m = ""]) [*line 394*]

Function Parameters:

- *array* **\$lines** contains the lines to convert to a properly indented string
- *string* **\$m** left margin for increased readability

get lines from an array in a single properly indented string

This is a workhorse to convert an array of lines to a properly indented block of text.

string function Theme::get_logo([\$m = ""]) [*line 509*]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct an image tag with the area logo

This constructs HTML-code that displays the logo.

- **TODO** should we take path_info into account here too???? how about /area/aaa/node/nnn instead of /aaa/nnn???

string function Theme::get_menu([\$m = "], [\$menu_id = NULL]) [*line 649*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *int* **\$menu_id** indicates where to start the menu (NULL means the first breadcrumb in top level menu)

construct the submenu starting at \$menu_id OR the first breadcrumb in the top level menu

this constructs an 'infinitely' nested set of submenus, starting at \$menu_id or at the first breadcrumb in the top level menu (if any). If there are no suitable nodes, an empty string is returned.

- **Uses** [Theme::show_tree_walk\(\)](#)

string function Theme::get_navigation([\$m = "], [\$textonly = FALSE]) [*line 622*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *bool* **\$textonly** forces a text-type link even when a navigation image is stipulated

construct a top level menu (navigation bar) as an unnumbered list (UL) of list items (LI)

this simply walks through the top level of the menu tree and creates a link for each node.

string function Theme::get_popups(\$messages, [\$m = ""]) [*line 451*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *array* **\$messages** @messages a collection of message to display via alert()

construct javascript alerts for messages

This constructs a piece of HTML that yields 0 or more calls to the javascript alert() function, once per message. If no messages need to be displayed an empty string is returned.

bool/array function Theme::get_properties(\$theme_id, \$area_id) [*line 970*]

Function Parameters:

- *int* **\$theme_id**
- *int* **\$area_id**

retrieve configuration parameters for this combination of theme and area

- Usedby [Theme::get_properties\(\)](#)
- Uses [Theme::get_properties\(\)](#)

string function Theme::get_quickbottom([\$m = ""]) [*line 549*]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct a list of quicklinks for bottom of page (if any)

(see also [get_quicktop\(\)](#)).

- Uses [Theme::get_quicklinks\(\)](#)

string function Theme::get_quicklinks(\$m, \$quick_section_parameter, [\$separator = "]) [*line 574*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *string* **\$quick_section_parameter** the name of the property that holds the quicklinks section
- *string* **\$separator** separates individual items in the list

workhorse for constructing list of quicklinks

This creates HTML-code for links that can be displayed at the top/bottom of the page. These links are the pages (but not subsections) defined in the quicktop_section_id or quickbottom_section_id in \$this->config.

Note that this array may or may not exist and also that the section may or may not exist and that the section may or may not contain any visible pages. Mmm, that's a lot of may/maybenot's...

Also note that these links are always displayed as text, even if a graphics image is defined in the corresponding node. The contents of the section can be found in \$this->tree. If there are two or more links, they are separated with \$separator (default "");

- Usedby [ThemeRosalina::get_quickbottom\(\)](#)
- Usedby [ThemeCornelia::get_quicktop\(\)](#)
- Usedby [Theme::get_quickbottom\(\)](#)
- Usedby [Theme::get_quicktop\(\)](#)

string function Theme::get_quicktop([\$m = "]) [*line 536*]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct a list of quicklinks for top of page (if any)

(see also [get_quickbottom\(\)](#)).

- Uses [Theme::get_quicklinks\(\)](#)

string function Theme::node2anchor(\$node_record, [\$attributes = NULL], [\$textonly = FALSE]) [*line 1067*]

Function Parameters:

- *array* **\$node_record** the node record to convert
- *array* **\$attributes** optional attributes to add to the HTML A-tag
- *bool* **\$textonly** if TRUE, no clickable images will be returned

construct an anchor from a node record

This constructs an array with key-value-pairs that can be used to construct an HTML anchor tag. At least the following keys are created in the resulting array: 'href', 'title' and 'anchor'. The latter is either the text or a referent to an image that is supposed to go between the opening A-tag and closing A-tag. Furthermore an optional key is created: target. The contents of the input array \$attributes is merged into the result.

If the parameter \$textonly is TRUE the key 'anchor' is always text. If \$textonly is NOT TRUE, the 'anchor' may refer to an image.

Note that the link text is always non-empty. If the node record has an empty link_text, the word 'node' followed by the node_id is returned. (Otherwise it will be hard to make an actual clickable link).

Note that we attempt to create 'friendly' URLs, ie. URLs that look very much like a plain path, e.g. `http://www.exemplum.eu/index.php/3/Information_about_the_school.html` rather than `http://www.exemplum.eu/index.php?node=3` When bookmarking a page, the part 'Information_about_the_school.html' makes it easier to recognise the bookmark than when it is just some number. Choice for friendly URLs is made in the global (site) configuration.

- Uses [was_node_url\(\)](#)

`void function Theme::queue_alert($message, [$username = "])` [line 1199]

Function Parameters:

- *string* **\$message** the message to add to the buffer of qualifying alert accounts
- *string* **\$username** (optional) the name of the user that initiated the action

add \$message to alerts watching this page

service routine for modules that are 'interactive', used in case data is added somehow, e.g. a visitor adds a post to an althing. In that case the module can decide to add an alert like
`$theme->queue_alert('New post nnn in althing', 'anonymous')`

`void function Theme::send_headers()` [line 188]

send collected HTTP-headers to user's browser

This sends the headers that still need to be sent. These are collected in the array `$this->http_headers`. If headers are already sent, this fact is logged (and the collected headers are not sent).

`void function Theme::send_output()` [line 214]

send collected output to user's browser

This first sends any pending HTTP-headers and subsequently outputs the page that is constructed by `$this->get_html()`. However, if `$silent_mode` is `TRUE`, we don't send anything. This allows for modules to 'get rid' of the navigation etc. and handle the I/O in the module (example: confab-module).

`void function Theme::set_preview_mode($is_preview_mode)` [line 957]

Function Parameters:

- *bool* **\$is_preview_mode** `TRUE` enables preview mode, `FALSE` disables it

set the preview mode

this sets the preview mode of the page currently being built. If it is set to `TRUE`, all internal URLs (such as those pointing to a node in the breadcrumb trail or in menu items) will be equal to `'#'` which makes it more or less impossible to leave the current page because a

bare '#' is considered an unnamed fragment and so no new page is loaded when the link is clicked; just the thing we need.

string function Theme::show_tree_walk([\$m = ""], \$subtree_id) [line 687]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *int* **\$subtree_id** indicates where to start this (sub)menu

workhorse for constructing recursive menu (walk the tree) along the breadcrumb trail

this constructs nested (sub)menus along the breadcrumb trail. The effect is that the (sub)menus that lead to the current page (\$this->node_id) are 'opened' whereas the other submenus are 'closed'. The (sub)menus are constructed in the form of nested UL's with LI's.

The level of recursion of the list items (LI) is indicated via class='levelNNN'. The type of item is indicated via class='page' or class='section'. Also, the item has an additional class='current' when it is part of the breadcrumb trail. Finally a current item also has the additional class 'activepage' or 'activesection' which makes the CSS easier.

The actual A-tag of the link only indicates being part of the breadcrumb trail via class='current'.

It is up to the style sheet to visualise these items taking all variants into account. Note that we only process visible pages and sections.

- Usedby [ThemeSophia::get_menu\(\)](#)
- Usedby [ThemeCornelia::get_menu\(\)](#)
- Usedby [ThemeRuta::get_menu\(\)](#)
- Usedby [Theme::get_menu\(\)](#)

Class TranslateTool

[line 56]

Methods to access properties of a language and modify translations

This class is used to manage languages and translations. The following functions are supplied

- add a new language
- edit the properties of an existing language (including active flag)
- add/edit translations of texts

The default action is to show a list of existing languages. From there the user can navigate to adding/editing language properties or manipulating translations in a particular language.

- **Package** wascore

TranslateTool::\$domains

array = array() [line 67]

- **Var** list of all language domains grouped by program, modules, themes and install

TranslateTool::\$languages

array = array() [line 64]

- **Var** list of all language records (including inactive ones), keyed with language_key

TranslateTool::\$output

object|null = NULL [line 58]

- **Var** collects the html output

TranslateTool::\$show_parent_menu

bool = FALSE [line 61]

- **Var** if TRUE the calling routing is allowed to use the menu area (e.g. show config mgr menu)

Constructor *void* function TranslateTool::TranslateTool(&\$output) [line 77]

Function Parameters:

- *object* **&\$output** collects the html output

construct a TranslateTool object

This initialises the TranslateTool and also dispatches the chore to do.

- **Uses** \$LANGUAGE
- **Uses** \$CFG

array function TranslateTool::a_param(\$chore, [\$language_key = NULL], [\$domain = NULL]) [line 992]

Function Parameters:

- *string* **\$chore** the next chore that could be done
- *string|null* **\$language_key** the language of interest or NULL if none
- *string|null* **\$domain** the full domain of interest or NULL if none

shorthand for the anchor parameters that lead to the translate tool

string function TranslateTool::code_highlight(&\$source, [\$highlight_on = ''], [\$highlight_off = '']) [line 1127]

Function Parameters:

- *string* **&\$source** the string that needs code highlighting
- *string* **\$highlight_on** is inserted before the code element that is highlighted
- *string* **\$highlight_off** is inserted after the code element that is highlighted

highlight code constructs in texts that are to be translated

this routine highlights the following code constructs:

- HTML-tags such as '' and ''
- Variables such as '{USERNAME}' and '{FILE}'
- Tildes in hotkeys such as '~Yes' and '~No'

All of these code elements are sandwiched between \$highlight_on and \$highlight_off. The HTML-tags are escaped using htmlspecialchars making it possible to actually display them as text (otherwise they might be rendered as actual code in the browser). The HTML-codes '

' and '<p>' receive special treatment: they are rendered as visible text and also as a newline.

Note: This assumes that all '{' are eventually followed by a '}'. As long as this is true, we can easily use a str_replace() to sandwich {VARIABLE} between highlights. If there is only a single '{' or '}' the highlights won't match. It could be a problem and if it is, the relevant code should iterate / chomp through the string with something like preg_match('/({[a-zA-Z0-9_]+})/', \$string, \$matches)

As an added bonus, sequences of two consecutive spaces are replaced with non-breakable spaces. This is handy for phrases that use spaces to indent text, e.g. in simple text-only email messages.

By using a reference we prevent the endless coping of (long) strings to the stack; this should save time & space.

bool function TranslateTool::diff_to_text(\$language_key, \$full_domain, &\$diff, &\$text) [*line 1349*]

Function Parameters:

- *string* **\$language_key** identifies the language
- *string* **\$full_domain** indicates the language domain
- *array* **&\$diff** contains all key-value-pairs for the modified translation
- *string* **&\$text** receives the complete sourcefile created from \$diff

convert an array with key-value-pairs to a php source file that can be included as a user translation

All key-value-pairs are converted to something like this:

```
...
$string['key'] = 'value';
...
```

We specifically use single quotes in order to prevent any variable expansion within the strings. We do escape embedded single quotes, naturally. Furthermore, some metadata is added to the top of the resulting file, including information about the creation time, the program version that was used, the version of the (English) source file on which this translation is based and finally information about the file version of the system strings which was used to diff against.

Note: If the file with these system strings do not exist (because the language is all new, indicated by a version 'v0', `_all_` strings are stored in the diff and thus in the user file. That file could be used as a new starting point for the new language in a next version of the program.

Note: we try very hard to defeat tricks with the contents of the metadata (i.e. we don't trust `_full_name` and `_email` to not contain tricks like `'*' followed by '/'` (which would prematurely end the comment in the header) etc.

array function TranslateTool::get_dialogdef_language([\$language_key = "]) [*line 729*]

Function Parameters:

- *string* **\$language_key** identifies the language to edit (empty string for add new language)

construct the language dialog (used for both add and edit)

this constructs a language dialog definition, maybe filled with data The main difference between dialogs for add and edit is that an existing language code (`$language_key`) cannot be changed; the corresponding field is shown in 'viewonly' mode. Another small difference is that an existing language cannot have itself as a parent language.

Note that we populate the 'edit' dialog with existing data from `$this->languages`.

array function TranslateTool::get_dialogdef_language_domain([\$language_key = "], [\$full_domain = "]) [*line 810*]

Function Parameters:

- *string* **\$language_key** identifies the language to edit
- *string* **\$full_domain** identifies the language domain

construct the translation dialog for selected language and domain

this constructs a translation dialog definition, filled with translations for language `$language_key` and domain `$full_domain`. The labels for the fields are derived from the English texts in `$full_domain`, in the order specified by the English file. If the English file

contains comments, these are added to the item too (to be displayed as additional information for the translator). The current translation of the \$full_domain is retrieved the usual way, via function t() (shorthand for \$LANGUAGE->get_phrase()) but without translating any variables (e.g. {VALUE}). Note that if a translation of a phrase does not exist in the target language, the get_phrase() routine will eventually yield the English translation (after trying the language parents first). Also note that the translated phrases could be retrieved from a user file (ie. a file from \$CFG->datadir/languages/\$language_key/\$full_domain).

- **TODO** try to figure this out: when the delimiter in \$name was a dot '.' \$_POST contained a '_' instead. WTF? (it seems that a colon works... for now)
- **Uses** \$USER
- **Uses** \$CFG

array function TranslateTool::get_domains() [*line 1020*]

return an ordered list of translation domains

this constructs a list of language domains, grouped by 'program','modules','themes' or 'install'. This array is the basis for validating full domains (in \$_POST'ed data) and also to construct a menu.

Note that we use the translations from the files themselves in the current language to construct this list. Every translatefile should have at least the string 'translatetool_title' and 'translatetool_description'. Currently the sort order is based on the (internal) name of the modules. This should do the trick for translators: the order of files to translate in the menu does not depend on the translation of the module- or theme-title. (In the page manager and elsewhere it may be different).

string function TranslateTool::get_icon_edit(\$language_key) [*line 969*]

Function Parameters:

- *string* **\$language_key**

construct a clickable icon to edit the properties of this language

- **Uses** \$WAS_SCRIPT_NAME

- **Uses \$USER**
- **Uses \$CFG**

array function TranslateTool::get_options_languages([\$skip_language_key = '']) [line 907]

Function Parameters:

- *string* **\$skip_language_key** suppress this language in list (language cannot be its own parent)

fetch a list of languages available as parent language

this constructs a list of languages that can be used as a list of parent language options in a listbox or radiobuttons.

void function TranslateTool::get_strings_system(\$language_key, \$full_domain, &\$string, &\$comment, \$languagekey, \$full_domein) [line 1175]

Function Parameters:

- *string* **\$languagekey** the two or three letter ISO 639 language code
- *string* **\$full_domein** the language domain of interest
- *array* **&\$string** receives the translations (this parameter must be called 'string')
- *array* **&\$comment** receives the comments (this parameter must be called 'comment')
- **\$language_key**
- **\$full_domain**

retrieve strings (translations) and comments from an official (system) translation file

This routine reads the system translations for \$language_key and \$full_domain from a file. For the translations of the main program we look for a single file in \$CFG->progdire/lanauages/\$language_key/. For modules, themes and addons we try two different locations: one within the moduel/theme/addon directory tree and subsequently in the generic directory \$CFG->progdire/lanauages/\$language_key/.

The names of modules/themes/addons are derived by stripping the 2-character prefix (m_, t_ or a_) from the full domain.

Translations for the installer are searched for in the /program/install/lanauages tree.

int function TranslateTool::guess_row_count(&\$text, [\$maximum = 15]) [*line 1081*]

Function Parameters:

- *string* **&\$text** the string to analyse
- *int* **\$maximum** the maximum value this routine returns

try to calculate a reasonable number of textarea rows based on the contents of \$text

By using a reference we prevent the endless coping of (long) strings to the stack; this should save time & space.

void function TranslateTool::languages_overview() [*line 175*]

display list of languages with edit icons and an option to add a language

this constructs the languages overview: a link to add a language, followed by a list of languages based on the languages in the database. Every language has an icon through which the properties of the language can be modified, including setting/resetting the active flag. (Only active languages can be used on the website and in the CMS). Note that we use `_all_` languages here, including inactive ones.

Note that the calling routine (the tools manager) is allowed to display a menu because we set the parameter `show_parent_menu` to `TRUE` here.

The constructed list looks something like this:

```
    Add a language
[E] Deutsch (de) (inactive)
[E] English (en)
[E] Nederlands (nl)
...
```

The clickable icons [E] lead to the Edit language properties. The clickable titles lead to the actual translations The clickable link 'Add an area' leads to the add new language dialog.

- **TODO** should we add a paging function to the (perhaps looooong) list of languages?
- **Uses** `$WAS_SCRIPT_NAME`
- **Uses** `$CFG`
- **Uses** `$USER`

void function TranslateTool::language_add() [line 231]

present the language dialog where the user can enter properties for a new language

this displays a dialog where the user can enter the properties of a new language. These properties are:

- name (expressed in the language itself)
- key (2- or 3-letter code, presumably based on ISO 639-1 or ISO 639-2)
- parent_key (this language is based on which existing language)
- active flag

The new language is saved via performing the 'chore'
TRANSLATETOOL_CHORE_LANGUAGE_SAVE_NEW.

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER

void function TranslateTool::language_edit() [line 252]

show the language edit dialog

display a dialog where the user can modify language properties. we re-use the routine that created the add language dialog.

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER

void function TranslateTool::language_save() [line 427]

validate and save modified data to database

this saves data from the edit language dialog if data validates. If the data does NOT validate, the edit screen is displayed again otherwise the languages overview is displayed again.

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER
- **Uses** \$LANGUAGE
- **Uses** \$CFG

void function TranslateTool::language_savenew() [line 300]

save the newly added language to the database

This saves the essential information of a new language to the database, using sensible defaults for the other fields. Also, a data directory is created in \$CFG->datadir

If something goes wrong, the user can redo the dialog, otherwise we return to the languages overview, with the newly added language in the list, too.

Apart from the standard checks the following checks are done:

- the language key should be an acceptable directory name
- the language key should be lowercase
- the language key should not exist already (in \$this->languages)
- the directory should not yet exist
- the directory must be created here and now

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$USER
- **Uses** \$LANGUAGE
- **Uses** \$CFG

bool function TranslateTool::put_strings_userfile(\$language_key, \$full_domain, &\$diff) [line 1283]

Function Parameters:

- *string* **\$language_key** identifies the language to save
- *string* **\$full_domain** indicates which language domain needs to be saved
- *array* **&\$diff** contains all key-value-pairs for the modified translation

save new or changed translations to a file under CFG->datadir/languages

array function TranslateTool::render_translation_dialog(\$href, &\$dialogdef, [\$method = 'post'], [\$attributes = '']) [line 1228]

Function Parameters:

- *string* **\$href** the target of the HTML form
- *array* **&\$dialogdef** the array which describes the complete dialog
- *string* **\$method** method to submit data to the server, either 'post' or 'get'
- *string/array* **\$attributes** holds the attributes to add to the form tag

render a translation dialog based on a dialog definition

This routine looks a bit like the generic [dialog_quickform\(\)](#). The differences are:

- we show a comment (if any) in a box before label and input
- the labels don't have hotkeys based on tildes at all (except the submit buttons)
- comments and labels are wrapped in separate div's especially for the occasion

We do take any errors into account: fields with errors are displayed using the additional error class (which shows a label completely in red to indicate the error).

- Uses [html_form\(\)](#)

void function TranslateTool::show_domain_menu(\$language_key, [\$current_domain = '']) [line 934]

Function Parameters:

- *string* **\$language_key** the language currently being edited
- *string* **\$current_domain** the currently selected language domain (used to emphasize the option in the menu)

display the domain menu via \$this->output

This displays a clickable menu on in the menu area on the left of the screen.

bool function TranslateTool::show_parent_menu() [line 132]

allow the caller to use the menu area (or not)

this routine tells the caller if it is OK to use the menu area (TRUE returned) or not (FALSE returned).

bool function TranslateTool::submit_diff_to_project(\$language_key, \$full_domain, &\$diff) [line 1408]

Function Parameters:

- *string* **\$language_key** identifies the language to submit
- *string* **\$full_domain** indicates which language domain needs to be submitted
- *array* **&\$diff** contains all key-value-pairs for the modified translation

send new or changed translations back to the project

This sends an e-mail back to the project with the translation. We do so in the form of an attachment, but with a 'safe' extension (.bin rather than .php). This means that we will be able to traverse any firewalls and spamfilters and malware detectors.

The `_notes` are used as the body of the message, the file is attached.

Note that we send a copy of the message to the site itself (either the from-address or the reply-to-address).

void function TranslateTool::translation_edit() [line 507]

show an edit dialog with phrases from \$full_domain in \$language_key

After some sanity checking this routine shows a dialog where the user can edit translations for the selected language and domain. Note that this could be a huge dialog, depending on the size of the language domain ('admin' is notoriously large). Sending this routine to the browser can take some time.

void function TranslateTool::translation_save() [line 568]

save the modified translations in a file in the tree CFG->datadir/languages/

this routine validates the dialog data and attempts to save the changes in the file `$full_domain` in the directory `CFG->datadir/languages/$language_key/`. Also, we may need to send a message to the Website@School project with our changes (depending on the flag `_submit`).

Class Useraccount

[line 254]

Methods to access properties of the account of the logged in user

This deals mainly with retrieving information about the user that is currently logged in. There is one exception: a user that is NOT logged in can still have a \$USER object, but there are no privileges in that case. The special user_id in that case is 0.

The constructor reads the important data from the database. This includes things like the full name of the user and the email address. This information is stored in the object and can be used, e.g. \$USER->email. This information is basically copied from the table 'users'.

Furthermore, any properties for this user are retrieved from the table 'users_properties'. All properties are stored in an array. These can be used directly via \$USER->properties['foobar'].

Access Control

Finally we deal with access control. This has become quite complex but still manageable (I hope). There are six tables dealing with acl's:

- acls: site-wide permissions for jobs, intranet, modules and nodes
- acls_areas: permissions for intranet, modules and nodes at the area level
- acls_nodes: permissions for modules and nodes at the node level
- acls_modules: permissions for modules at the site level
- acls_modules_areas: permissions for modules at the area level
- acls_modules_nodes: permissions for modules at the node level

The user has at least one associated ACL: the acl_id field in the user record. Additional ACLs are associated with the user via group memberships. All ACLs are integer bitmasks where a '1' grants a permission for something and a '0' denies permission.

All bits '0' is a special case: this is the default (nothing allowed) and hence does not have to be stored: the mere non-existence of permissions implies no permissions.

All bits '1' is also a special case, dubbed 'ROLE_GURU'. If an ACL has this value, it means that all current (and future) permissions are granted. A user with ROLE_GURU can do anything.

Of the six tables, only the first one (acls) is read immediately in the constructor. The others are read on demand. This is done by initially setting the corresponding cache variable to NULL. If the table has been read, the variable will always be of type 'array', even though that array may be empty (indicating no permissions).

The permissions from the ACLs are combined between the user's acl and the optional group-acls. Only the combination of user and group permissions is cached in order to save space. This is done by OR'ing the permission bits. Note that the condition all bits '0' is not stored, also to save space.

There are some functions to test for individual permissions:

- `has_site_permissions()`
- `has_area_permissions()`
- `has_node_permissions()`
- `has_module_site_permissions()`
- `has_module_area_permissions()`
- `has_module_node_permissions()`
- `has_job_permissions()`
- `has_intranet_permissions()`

Example: in order to determine whether a user has access to the intranet in area #2, the following could be used:

```
$area_id = 2;
if ($USER->has_intranet_permissions(ACL_ROLE_INTRANET_ACCESS,$area_id)) {
    ....
}
```

The effect of this call is as follows. First the routine checks the (already cached) intranet-permissions at the site level. If access is granted at the site level, there is no need to look any further because obviously this user has access to this intranet (private area) and all other current and future intranets. If not, the routine looks at intranet permissions at the area level. The first time this will trigger reading and caching the table for area-level permissions. In this case (intranet-access), the area-level permissions provide the definitive go/nogo for this user (there is no point in having intranet-access-permissions at the node level).

Note that the 'lower' ACL is only checked if the 'higher' does not provide answers. This saves unnecessary trips to the database.

Note that this works much the same for the other `has_xxx_permissions()`: first the site-level is tried, then the area-level and finally the node-level (when applicable).

ACLs for modules

Access to the CMS itself is fairly fine-grained. The permissions are stored in the fields 'permissions_nodes' in the tables 'acls' (site-level), 'acls_areas' (area-level) and 'acls_nodes' (node-level). These permissions basically deal with the page manager (the piece de resistance of the whole system).

However, there are modules that can be linked to nodes, e.g. a chat or a forum or an agenda which also require authorised users and permissions. These permissions are stored in three tables: 'acls_modules' (site-level), 'acls_modules_areas' (area-level) and 'acls_modules_nodes' (node-level). This works pretty much the same as the permissions for the CMS itself, be it that there is an extra parameter, namely the `module_id`.

Once again, the permissions are only read when necessary. I.e., if the site-level already grants a permission, the area and node level are not read from the database. This saves time and space.

Roles and permissions

Permissions are individual flags that allow or disallow a certain feature, e.g. 'adding a page to a section'. In order to keep these permissions manageable groups of permissions are combined yielding a limited number of 'roles'. A 'role' is a combination of 1 or more permission bits. Assigning permissions (in the user account manager) is done by assigning these 'roles' to a user, either sitewide, areawide or per node. These roles are dubbed sitemaster, areamaster, sectionmaster, pagemaster and contentmaster. The 'higher' roles incorporate the 'lower' roles: permissions of a sectionmaster include those of a pagemaster and a contentmaster.

- **Package** wascore

Useraccount::\$scls

```
array = array('permissions_jobs' => ACL_ROLE_NONE,
              'permissions_intranet' => ACL_ROLE_NONE,
              'permissions_modules' => ACL_ROLE_NONE,
              'permissions_nodes' => ACL_ROLE_NONE) [line 280]
```

- **Var** \$scls caches site-level permissions, keyed by [\$field]

Useraccount::\$scls_areas

```
null|array = NULL [line 286]
```

- **Var** \$scls_areas caches area-level permissions, keyed by [\$area_id][\$field]

Useraccount::\$scls_modules

```
null|array = NULL [line 292]
```

- **Var** \$scls_modules site-level modules permissions, keyed by [\$module_id]

Useraccount::\$scls_modules_areas

```
null|array = NULL [line 295]
```

- **Var** \$scls_modules_areas area-level modules permissions, by [\$module_id][\$area_id]

Useraccount::\$scls_modules_nodes

null|array = NULL [line 298]

- **Var** \$scls_modules_nodes node-level modules permissions, by [\$module_id][\$node_id]

Useraccount::\$scls_nodes

null|array = NULL [line 289]

- **Var** \$scls_nodes caches node-level permissions, keyed by [\$node_id][\$field]

Useraccount::\$acl_id

int = 0 [line 274]

- **Var** \$acl_id identifies the main acl for this user

Useraccount::\$area_permissions_from_nodes

array|null = NULL [line 313]

- **Var** cache for admin permissions based on node permissions

Useraccount::\$editor

string = [line 304]

- **Var** \$editor the user's preferred editor (empty means system default from \$CFG->editor)

Useraccount::\$email

string = [line 265]

- **Var \$email**

Useraccount::\$full_name

string = [line 262]

- **Var \$full_name**

Useraccount::\$is_logged_in

bool = FALSE [line 310]

- **Var \$is_logged_in** TRUE if user is logged in, FALSE otherwise

Useraccount::\$language_key

string = [line 268]

- **Var \$language_key**

Useraccount::\$path

string = [line 271]

- **Var \$path** directory holding personal data files relative to "{\$CFG->datadir}/users/"

Useraccount::\$properties

array = array() [line 301]

- **Var \$properties**

Useraccount::\$related_acls

array = array() [line 277]

- **Var \$related_acls** holds acl_id -> groupname/capacity pairs related to this user

Useraccount::\$skin

string = [line 307]

- **Var \$skin** the preferred skin for this user

Useraccount::\$username

string = [line 259]

- **Var \$username**

Useraccount::\$user_id

int = 0 [line 256]

- **Var** \$user_id

Constructor *void* function Useraccount::Useraccount([\$user_id = 0]) [*line 330*]

Function Parameters:

- *int* **\$user_id** identifies data from which user to load, 0 means no user/a passerby

get pertinent user information in core

Note: We used to have a bool named 'high_visibility' in both the users table and this class. That changed with version 0.90.4 (April 2012) and we now have a field and variable 'skin' which is a varchar(20). The values were mapped as follows: high_availability=FALSE -> skin='base' and high_availability=TRUE -> skin='textonly'. The extra test for the existence of \$record['skin'] was necessary for the case where the user wanted to upgrade from 0.90.3 to 0.90.4 where 'skin' replaced 'high_visibility'.

array function Useraccount::fetch_acls_from_table(\$table) [*line 678*]

Function Parameters:

- *string* **\$table** name of the table which holds the acls

retrieve acl-data from table into a sparse array

bool function Useraccount::has_area_permissions(\$mask, \$area_id, [\$field = 'permissions_nodes']) [*line 421*]

Function Parameters:

- *int* **\$mask** bitmap of OR'ed permissions to test for
- *int* **\$area_id** which area to test
- *string* **\$field** name of permissions to check (default 'permissions_nodes')

determine user's permissions for an area

this looks at the area-level permissions for manipulating nodes and areas. However, we first look at the site-level permissions. If those already satisfy the request, we return immediately. If not, the permissions are fetched from the table acls_areas or from the cached data. We only fetch the data if it is really necessary.

bool function Useraccount::has_intranet_permissions(\$mask, \$area_id) [*line 587*]

Function Parameters:

- *int* **\$mask** bitmap of OR'ed permissions to test for
- *int* **\$area_id** which area to test

determine user's permissions for an intranet area

this looks at the area-level permissions for intranet areas.

bool function Useraccount::has_job_permissions(\$mask) [*line 571*]

Function Parameters:

- *int* **\$mask** bitmap of OR'ed permissions to test for

determine user's permissions for a job

bool function Useraccount::has_module_area_permissions(\$mask, \$module_id, \$area_id) [*line 513*]

Function Parameters:

- *int* **\$mask** bitmap of OR'ed permissions to test for
- *int* **\$area_id** which area to test
- *int* **\$module_id** module_id identifies the module we are considering

determine user's permissions for a module at the area level

this looks at the area-level permissions for manipulating nodes and areas. However, we first look at the site-level permissions. If those already satisfy the request, we return immediately. If not, the permissions are fetched from the table `acls_modules_areas` or from the cached data. We only fetch the data if it is really necessary.

bool function Useraccount::has_module_node_permissions(\$mask, \$module_id, \$area_id, \$node_id) [*line 544*]

Function Parameters:

- *int* **\$mask** bitmap of OR'ed permissions to test for

- *int* **\$area_id** which area to test
- *int* **\$node_id** which node to test
- *int* **\$module_id** module_id identifies the module we are considering

determine user's permissions for a module at the node level

- **TODO** FixMe: we need to take the parent nodes into account too!

bool function Useraccount::has_module_site_permissions(\$mask, \$module_id) [line 479]

Function Parameters:

- *int* **\$mask** bitmap of OR'ed permissions to test for
- *int* **\$module_id** identifies the module we are considering

determine user's permissions for a module at the site-level

this looks at the site-level permissions for manipulating modules. The permissions are cached from the table acls_modules.

bool function Useraccount::has_node_permissions(\$mask, \$area_id, \$node_id, [\$field = 'permissions_nodes']) [line 450]

Function Parameters:

- *int* **\$mask** bitmap of OR'ed permissions to test for
- *int* **\$area_id** which area to test
- *int* **\$node_id** which node to test
- *string* **\$field** name of permissions to check (default 'permissions_nodes')

determine user's permissions for a node within an area

- **TODO** FixMe: we need to take the parent nodes into account too!

bool function Useraccount::has_site_permissions(\$mask, [\$field = 'permissions_nodes']) [*line 396*]

Function Parameters:

- *int* **\$mask** bitmap of OR'ed permissions to test for
- *string* **\$field** name of permissions to check (default 'permissions_nodes')

determine user's permissions for the site-level

this looks at the site-level permissions for manipulating nodes and areas etc. The permissions are cached from the table acs.

bool function Useraccount::is_admin() [*line 604*]

determine whether the user has administrator privilege

If this user has access to the admin startcenter, she is considered an administrator. Further access depends on the other bits in the job permissions, but at least she is allowed to enter the system via admin.php.

bool function Useraccount::is_admin_pagemanager(\$area_id) [*line 628*]

Function Parameters:

- *int* **\$area_id** the area to examine

determine whether the user has administrator privilege for pagemanager

This routine determines whether a user has any privileges at all for the page manager. This is true in the following cases:

- the user has sitewide permissions that belong to one of the roles contentmaster, pagemaster, sectionmaster or areamaster, OR
- the user has areawide permissions for one of those roles, OR
- the user has permissions for one of those roles in at least one node in the requested area.

The calculations in the third case are cached for all areas.

string function Useraccount::where_acl_id([\$field = 'acl_id']) [*line 760*]

Function Parameters:

- *string* **\$field** identifies the fieldname to check

a convenient routine to construct a selection of acls

this constructs a where clause of the form '(acl_id = 1) OR (acl_id = 2) OR (acl_id = 3)'

Class UserManager

[*line 40*]

User management

- **Package** wascore
- **TODO** Perhaps this class should be merged with the GroupManager class because there is a lot of overlap. Mmmmm.... maybe in a future refactoring operation.

UserManager::\$output

object|null = NULL [*line 42*]

- **Var** collects the html output

UserManager::\$show_parent_menu

bool = FALSE [*line 45*]

- **Var** if TRUE the calling routing is allowed to use the menu area (e.g. show account mgr menu)

UserManager::\$users

array = array() [line 48]

- **Var** used to cache user records keyed by user_id

Constructor *void* function UserManager::UserManager(&\$output) [line 57]

Function Parameters:

- *object* **&\$output** collects the html output

construct a UserManager object

This initialises the UserManager and also dispatches the task to do. This also loads the loginlib: we need that in order to manipulate the user password.

array|bool function UserManager::areas_expand_collapse(\$areas_open, \$area_id) [line 2227]

Function Parameters:

- *array|bool* **\$areas_open** current state of indicator(s) for 'open' and 'closed' areas
- *int|null* **\$area_id** the area to expand/collapse or NULL if nothing needs to be done

manipulate the current state if indicator(s) for 'open' and 'closed' areas

this manipulates the current state of 'open' and 'closed' areas in \$areas_open. If \$area_id is NULL, we don't have to do anything but simply return the current state. If \$area_id is 0 (zero), we need to toggle all areas at once (area_id = 0 implies the site level toggle) If \$area_id is an integer, it is assumed to be a valid area_id and that area should be toggled.

array function UserManager::a_params([\$task = NULL], [\$user_id = NULL], [\$module_id = NULL]) [line 1700]

Function Parameters:

- *string|null* **\$task** the next task to do or NULL if none
- *int|null* **\$user_id** the user of interest or NULL if none

- *int|null* **\$module_id** the module of interest or NULL if none

shorthand for the anchor parameters that lead to the user manager

bool/int function UserManager::calc_acl_id(\$user_id) [*line 2172*]

Function Parameters:

- *int* **\$user_id** identifies the user record of interest

determine the acl_id for user user_id

bool function UserManager::delete_user_records(\$user_id) [*line 1401*]

Function Parameters:

- *int* **\$user_id** the key to the user account to delete

remove all records relating to a single acl_id from various acl-tables

this bluntly removes all records from the various user-related tables for user \$user_id. Whenever there's an error deleting records, the routine bails out immediately and returns FALSE. If all goes well, TRUE is returned. Any errors are logged, success is logged to DEBUG-log.

Note that the order of deletion is important: we must first get rid of the foreign key constraints.

array function UserManager::get_dialogdef_add_user() [*line 1882*]

construct the add userdialog

array function UserManager::get_dialogdef_add_usergroup(\$user_id) [*line 1314*]

Function Parameters:

- *int* **\$user_id** limit the options to groups this user is NOT already a member of

construct a dialogdef for selecting a group/capacity

array function UserManager::get_dialogdef_confirm_delete() [*line 2119*]

construct the edit user dialog

bool|array function UserManager::get_dialogdef_edit_user(\$user_id) [*line 1968*]

Function Parameters:

- *int* **\$user_id** indicates which user to edit

construct the edit user dialog

- **Uses** \$LANGUAGE

string function UserManager::get_fname(\$item) [*line 2185*]

Function Parameters:

- *array* **\$item** contains definition of a single field in a dialog

shorthand for the first readable name in a dialogdef item

string function UserManager::get_icon_delete(\$user_id) [*line 1814*]

Function Parameters:

- *int* **\$user_id** the user to delete

construct a clickable icon to delete this user

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

string function UserManager::get_icon_edit(\$user_id) [*line 1837*]

Function Parameters:

- *int* **\$user_id** the user to edit

construct a clickable icon to edit the properties of this user

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

string function UserManager::get_icon_groupdelete(\$user_id, \$group_id) [*line 1861*]

Function Parameters:

- *int* **\$user_id** the group to delete
- **\$group_id**

construct a clickable icon to delete a membership from this user

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

int function UserManager::get_num_user_records(\$group_id) [*line 1777*]

Function Parameters:

- *int* **\$group_id** which group needs to be counted

calculate the total number of users in a specific group

this calculates the total number of users in group \$group_id. If \$group_id equates to GROUP_SELECT_ALL_USERS, the grand total is returned, if it equates to

GROUP_SELECT_NO_GROUP the number of users without a group is calculated.

- **Uses \$DB**

array function UserManager::get_options_available_groups_capacities(\$user_id) [line 1361]

Function Parameters:

- *int* **\$user_id** the user to which this list of available groups applies

construct a list of groups still available for this user

this constructs an array with available groups/capacities for the user \$user_id. If the user is already a member of all available groups or there are no groups at all, the list consists of a single option 'No groups available'.

The values in this list are constructed from the primary key values of the underlying groups_capacities table. These two numbers (group_id and capacity_code) are separated with a colon ':' to make it easier to parse once we are to save the values (in the table users_groups_capacities).

The SQL-statement looks quite complex. What it does is using the table groups_capacities as a starting point for _all_ valid (ie capacity_id != CAPACITY_NONE) combinations of group and capacity. By left-joining the table users_groups_capacities with a very specific ON-clause, and leaving out the column capacity_code, the resulting list consists of all combinations of group and capacity but without any entries that have a group of which the user is already a member, no matter what capacity. In other words: if a user is already a member of a group with capacity A, this user cannot be member of the same group with capacity B. Finally, the table groups is used to retrieve the group information such as the groupname and the active-flag.

The resulting list is ordered by groupname and subsequently by the sort_order of the capacity_code. However, inactive groups are sorted after the active groups so they appear near the bottom of the list.

array function UserManager::get_user_names(\$user_id) [line 2138]

Function Parameters:

- *int* **\$user_id** identifies the user of interest

shortcut to retrieve the username and full name of the selected user

bool/array function UserManager::get_user_record(\$user_id, [\$forced = FALSE]) [*line 2197*]

Function Parameters:

- *int* **\$user_id** identifies the user record
- *bool* **\$forced** if TRUE unconditionally fetch the record from the database

retrieve a single user's record possibly from the cache

array/bool function UserManager::get_user_records(\$group_id, \$limit, \$offset) [*line 1732*]

Function Parameters:

- *int* **\$group_id** selection for users
- *int* **\$limit** maximum number of records to retrieve
- *int* **\$offset** number of records to skip in result set

retrieve (a selection of) all user records from the database

this retrieves a subset of all existing user accounts from the database. The selection depends on the value of \$group_id:

- `$group_id == GROUP_SELECT_ALL_USERS (-1)`: all users ordered by active, username
- `$group_id == GROUP_SELECT_NO_GROUP (0)`: all users without a group, ordered by active, username
- otherwise: users in group \$group_id, ordered by active, username

Note that in the first two cases there is no capacity, in the third case every user has capacity relating to the specified group.

bool function UserManager::has_job_permission(\$user_id, \$job) [*line 2156*]

Function Parameters:

- *int* **\$user_id** group to check
- *int* **\$job** job a bitmask indicating a particular job

determine whether a user has permissions for a particular job

this determines whether this user has permissions to access the specified job, e.g. do they have access to the page manager. If so, we can display the menu option, otherwise we can suppress it and keep the menu clean(er).

void function UserManager::show_breadcrumbs_adduser() [line 1683]

display breadcrumb trail that leads to the add new user dialog

- **Uses** [UserManager::show_breadcrumbs_overview\(\)](#)
- **Uses** \$WAS_SCRIPT_NAME;

void function UserManager::show_breadcrumbs_overview() [line 1639]

display breadcrumb trail that leads to users overview screen

- **Usedby** [UserManager::show_breadcrumbs_adduser\(\)](#)
- **Usedby** [UserManager::show_breadcrumbs_user\(\)](#)
- **Uses** \$WAS_SCRIPT_NAME;

void function UserManager::show_breadcrumbs_user(\$user_id) [line 1666]

Function Parameters:

- *int* **\$user_id** the user of interest

display breadcrumb trail that leads to the edit user dialog

- **Uses** [UserManager::show_breadcrumbs_overview\(\)](#)
- **Uses** \$WAS_SCRIPT_NAME;

void function UserManager::show_menu_overview(\$group_id) [line 1559]

Function Parameters:

- *int|null* **\$group_id** identifies the current selection

display a menu showing groups of users (if any) + corresponding breadcrumb trail

this constructs a list of links allowing for a quick selection of a subset of users This looks a little like this:

All users (66)
No group (5)
faculty (14)
grade12 (7)
...
webmasters (2)

The indication of the current selection in the menu is based on \$group_id. Most of the time this is a genuine group_id. However, 'All users' and 'No group' are special cases:

- The value GROUP_SELECT_ALL_USERS (-1) cannot be a genuine group_id because these are always > 0.
- The value GROUP_SELECT_NO_GROUP (0) cannot be a genuine group_id because these are always > 0.

- **Uses** \$DB;
- **Uses** #AS_SCRIPT_NAME

void function UserManager::show_menu_user(\$user_id, [\$current_task = NULL], [\$current_module_id = NULL]) [line 1454]

Function Parameters:

- *int* **\$user_id** identifies the user
- *string* **\$current_task** the task to show highlighted

- `int $current_module_id` the current module to show highlighted

show the user menu with current option highlighted

this constructs the user menu. Only the relevant options are displayed (eg. if the user is not an admin, no pagemanager option is displayed).

`void function UserManager::show_parent_menu() [line 98]`

`void function UserManager::users_overview() [line 148]`

display a list of existing users and an option to add a user

This constructs the heart of the user manager: a link to add a user, followed by a list of links for deleting an modifying selected (see below) users. The list of users is ordered as follows. First the active users are displayed, an after that the inactive users are displayed. The sort order is based on the short name of the user.

Note that a selection is made of all user accounts, based on a choice the user makes from the menu (see [show_menu_overview\(\)](#)). This list to show is selected as follows:

- if the parameter 'group' is NOT set in `$_GET[]` and this is the 1st time, all users are listed (equivalent with `GROUP_SELECT_ALL_USERS`). If we are returning, the `$_SESSION` may contain another default group selection
- if the parameter 'group' is set to `GROUP_SELECT_ALL_USERS (-1)`, all users are listed
- if the parameter 'group' is set to `GROUP_SELECT_NO_GROUP (zero)`, all users without a group are listed
- if the parameter 'group' has another value, the users of that group are listed

The list of existing users is paginated, ie. if there are more than a screenfull, an additional paginator is displayed at the end of the list. The screen always starts with an add a user link though.

Note that the list of existing users shows the full name and the username in parenthese. If a 'real' group is selected (ie. not the collection of users without a group or all users), the capacity of that user in that group is also displayed.

Example: Amelia Cackle, a 'Principal' in the 'faculty' group, is displayed like this in the faculty group: Amelia Cackle (acackl) (Principal)

`void function UserManager::user_add() [line 277]`

present 'add user' dialog where the user can enter minimal properties for a new user

this displays a dialog where the user can enter the minimal necessary properties of a

new user. These properties are:

- name (e.g. 'hparkh')
- full name (e.g. 'Helen Parkhurst')
- a password
- an e-mail address
- the active flag

Other properties will be set to default values and can be edited later on by editing the user account.

The new user is saved via performing the task TASK_USER_SAVE_NEW

- **Uses** \$WAS_SCRIPT_NAME

void function UserManager::user_admin() [line 1197]

show a dialog for modifying admin permissions for a user

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

void function UserManager::user_delete() [line 771]

delete a user after confirmation

after some basic tests this either presents a confirmation dialog to the user OR deletes a user with associated acls and other records.

Note that this routine could have been split into two routines, with the first one displaying the confirmation dialog and the second one 'saving the changes'. However, I think it is counter-intuitive to perform a deletion of data under the name of 'saving'. So, I decided to use the same routine for both displaying the dialog and acting on the dialog.

Note that the (user)files should be removed before the account can be removed, see [userdir_is_empty\(\)](#). It is up to the user or the admin to remove those files.

A special test is performed to prevent users from killing their own account (which would immediately kick them out of admin.php never to be seen again).

- **TODO** since multiple tables are involved, shouldn't we use transaction/rollback/commit? Q: How well is MySQL suited for transactions? A: Mmmmm.... Which version? Which storage engine?

void function UserManager::user_edit([\$user_id = NULL]) [line 479]

Function Parameters:

- *int* **\$user_id** indicates which user to edit

present an 'edit user' dialog filled with existing data

this prepares the basic user properties dialog, based on the parameter \$user_id. If that is not specified, we read the \$user_id from \$_GET. (Currently only [user_savenew\(\)](#) and [user_save_basic\(\)](#) call us with \$user_id set).

void function UserManager::user_groupadd([\$user_id = NULL]) [line 972]

Function Parameters:

- **\$user_id**

present 'add membership' dialog

this displays a simple dialog where the user can add a membership to a user account, one at a time. Basically we show a picklist with all available group/capacity-combinations. Here "available" means:

- only groups of which the user is currently NOT a member
- only non-0 group/capacity-combinations that occur in the groups_capacities_table (capacity 0 implies: no capacity)

An additional feature is that the user can become member of inactive groups. However, these groups are sorted at the end of the picklist.

\$param int \$user_id identifies the user to edit

void function UserManager::user_groupdelete() [line 1109]

end the group membership for the selected user

- **Uses \$DB**

void function UserManager::user_groups() [line 895]

present an overview of group memberships for the specified user

this constructs a link to add a membership to the user account and a list of existing memberships, if any, including a delete button per membership.

The SQL-query retrieves the list of existing memberships from the database, ordered by the short groupname. The data is validated by joining to the table groups_capacities. If for some reason there exists an invalid combination of group_id and capacity_code in users_groups_capacities table, it will not show up in the list here.

Note that it is currently not possible to change a users' group membership, i.e. you cannot promote a user from 'Member' to 'Chair' for a group: you have to delete the group membership first, and subsequently add it again with the correct capacity.

void function UserManager::user_groupsave() [line 1010]

save the new group/capacity for the selected user

this adds a record to the users_groups_capacities table, indicating the group membership and the corresponding capacity for the user.

- **Uses \$WAS_SCRIPT_NAME**

void function UserManager::user_intranet() [line 1155]

show a dialog for modifying intranet permissions for a user

- **Uses \$WAS_SCRIPT_NAME**
- **Uses \$CFG**

void function UserManager::user_pagemanager() [line 1238]

show a dialog for modifying page manager permissions for a user

- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

void function UserManager::user_save() [line 508]
save edited user data to the database

data function UserManager::user_savenew() [line 301]
save a new user to the database

this saves a new user to the database. This involves at least two tables: a record in the users table with basic information and also a record with access control in the acls table.

- **TODO** shouldn't we end with the edit-user dialog rather than the users overview? that might make more sense...
- **TODO** maybe we should find a more elegant way to check a field for uniqueness
- **Uses** \$WAS_SCRIPT_NAME
- **Uses** \$CFG

void function UserManager::user_save_basic(\$user_id) [line 635]
Function Parameters:

- *int* **\$user_id** the account to save (pkey in users table)

save basic properties of user account

- **Uses** \$WAS_SCRIPT_NAME

Class Zip

[line 139]

Create simple and compatible ZIP-archives

With this class it is possible to create ZIP-archives that are compatible with the original PKZip 2.04g. This class does not provide a way to read ZIP-archives.

There are three possible options for the output:

- write the ZIP-archive directly to a file (OpenZipfile())
 - output ('stream') directly to the user's browser, including appropriate headers (OpenZipstream())
 - collect the output in a buffer in memory (OpenZipbuffer()).
There are two different ways to add to the ZIP-archive:
 - add a file from the filesystem (AddFile())
 - add data from memory as if it was a file (AddData())
- The ZIP-archive needs to be closed before it is useable (CloseZip()).

Special features:

- it is not necessary to manually add a directory to the ZIP-archive because all directories that lead to a file will be added automatically
 - both AddFile() and AddData() allow for on-the-fly (re)naming; i.e. the name of the file in the ZIP-archive can be different from the name of the file in the filesystem
 - it is possible to add a comment to an individual file
 - it is possible to add a comment to the ZIP-archive
- Limitations

This class might use a lot of memory when creating ZIP-archives, especially the ZIP_TYPE_BUFFER variant which eventually requires the size of the resulting ZIP-archive plus (worst case) the size of the largest file plus the size of the largest compressed file. This might be a problem with large files or many, many smaller files. A workaround could be to either stream the ZIP-archive directly (ZIP_TYPE_STREAM) or write to a file (ZIP_TYPE_FILE) because those variants only require the size of the largest file, the largest compressed file and the size of the central directory.

This class is not able to read ZIP-archives.

This class either stores a file as-is using the PKZIP 'Store' method or compresses the file using the 'Deflate' method. There is no support for other (more advanced) compression algorithms and no encryption is used.

References

I implemented this class using the following references.

[1] The ultimate definition of the ZIP-archive format as published by PKWare, Inc. See: <http://www.pkware.com/appnote.txt> or <http://www.pkware.com/support/zip-application-note>. I used version 6.3.2 which was published on 28 September 2007.

[2] RFC1950 - ZLIB Compressed Data Format Specification version 3.3, P. Deutsch, J-L. Gailly (May 1996), <http://www.faqs.org/rfcs/rfc1950>

[3] RFC1951 - DEFLATE Compressed Data Format Specification version 1.3, P. Deutsch (May 1996), <http://www.faqs.org/rfcs/rfc1951>

[4] Disk Operating System Technical Reference, IBM Corporation 1985, Chapter 5 (DOS Disk Directory).

[5] Official registration of the application/zip MIME-type: <http://www.iana.org/assignments/media-types/application/zip>

Examples

Typical usage of this class is as follows.

```
Example 1 - store 3 existing files in a ZIP-archive $zip = new Zip;
$zip->OpenFile("/tmp/test.zip");
$zip->AddFile("/tmp/foo.txt");
$zip->AddFile("/tmp/bar.txt");
$zip->AddFile("/tmp/baz.txt");
$zip->CloseZip();
```

```
Example 2 - store a chunk of data in a ZIP-archive in memory $zip_archive = "";
$data = "This is example-data that ends up in file QUUX.TXT";
$zip = new Zip;
$zip->OpenZipbuffer($zip_archive);
$zip->AddData($data,'QUUX.TXT');
$zip->CloseZip();
```

```
Example 3 - directly stream a file in a ZIP-archive and rename on the fly $zip = new Zip;
$zip->OpenStream('htdocs.zip');
$zip->AddFile("/var/www/index.html",'INDEX.HTM');
$zip->CloseZip();
```

All methods return TRUE on success or FALSE on failure. If the method failed, an (English) error message can be found in \$zip->Error.

- **Package** wascore

Zip::\$central_directory

array = array() [line 145]

- **Var** \$central_directory buffer for the central directory entries

This array is keyed by relative filename (both files and directories), no leading '/' though directories have a trailing '/'.

Zip::\$Error

string = [line 157]

- **Var** \$Error collects error messages if things go wrong

Zip::\$no_name_files

int = 0 [line 169]

- **Var** \$no_name_files is used to construct names for otherwise unnamed files

Zip::\$offset

int = 0 [line 148]

- **Var** \$offset always points to the next local file header in the ZIP-archive

Zip::\$zip_buffer

string = [line 166]

- **Var** \$zip_buffer reference to output buffer if \$zip_type is ZIP_TYPE_BUFFER

Zip::\$zip_comment

string = [line 154]

- **Var** \$zip_comment a file wide comment

Zip::\$zip_filehandle

null|resource = NULL [line 163]

- **Var** \$zip_filehandle handle on the zipfile output if \$zip_type is ZIP_TYPE_FILE

Zip::\$zip_path

string = [line 160]

- **Var** \$zip_path name of the zipfile if \$zip_type is ZIP_TYPE_FILE

Zip::\$zip_type

string = ZIP_TYPE_NONE [line 151]

- **Var** \$zip_type ZIP-archive destination: file, stream or buffer

Constructor `void` function `Zip::Zip()` [line 172]

constructor initialises all variables

`bool` function `Zip::AddData($data, [$filename = "], [$comment = "], [$timestamp = 0])` [line 317]

Function Parameters:

- `string $data` the data to add to the ZIP-archive
- `string $filename` the preferred name of the file in the ZIP-archive
- `string $comment` an optional comment for this specific file
- `int $timestamp` the unix timestamp to associate with the file

add data to the current ZIP-archive

This adds the data to the current ZIP-archive.

`bool` function `Zip::AddFile($path, [$filename = "], [$comment = "])` [line 276]

Function Parameters:

- `string $path` the full (absolute) name of the file to add to the ZIP-archive
- `string $filename` the preferred name of the file in the ZIP-archive (default is `$path`)
- `string $comment` an optional comment for this specific file

add the contents of an existing file to the current ZIP-archive

this reads the file `$path` into a buffer, and subsequently adds the data to the ZIP-archive.

`bool` function `Zip::CloseZip()` [line 344]

finish the ZIP-archive by outputting the central directory and closing output

this finishes the ZIP-archive by constructing and outputting the Central Directory and subsequently closing the output file (in case of `ZIP_TYPE_FILE`). The call to `CloseZip()` is necessary to create a complete ZIP-archive, including the Central Directory.

`string` function `Zip::dos_time_date($timestamp)` [line 675]

Function Parameters:

- *int* **\$timestamp** unix timestamp (seconds since 1970-01-01 00:00:00)

construct an MS-DOS time and date based on unix timestamp

this routine constructs a string of 2 x 2 bytes with the time and the date in the following format.

```
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
h h h h h m m m m m m x x x x x
```

hhhhh = hour, from 0 - 23 (5 bits)
 mmmmmm = minute, from 0 to 59 (6 bits)
 xxxxx = duoseconds, from 0 to 29 (5 bits)

```
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
y y y y y y y m m m m d d d d d
```

yyyyyyy = year offset from 1980, from 0 - 119 (7 bits)
 mmmm = month, from 1 to 12 (4 bits)
 dddd = day, from 1 to 31 (5 bits)

Note that the time resolution is 2 seconds whereas the unix timestamp has a 1 second resolution. This means that the seconds are rounded down. Also note that the specification [4] indicates that the maximum value for year offset is 119 which corresponds with 2099 rather than the maximum of 127 which would yield the year 2107.

string function Zip::make_suitable_filename(\$filename) [*line 622*]

Function Parameters:

- *string* **\$filename** name to analyse/massage

construct a suitable filename for use in ZIP-archive

this analyses and edits the string \$filename in such a way that a suitable name for use in a ZIP-archive remains. This means that:

- MS-DOS driverletters are removed from the path
- backslashes are replaced with slashes
- leading './' if any is removed
- a leading slash is removed

bool function Zip::OpenZipbuffer(&\$buffer, [\$comment = "]) [*line 252*]

Function Parameters:

- *string* **&\$buffer** a pointer to the buffer where we can write the ZIP-archive
- *string* **\$comment** an optional comment to include in the ZIP-archive

prepare the user supplied buffer for subsequent ZIP-archive data

bool function Zip::OpenZipfile(\$path, [\$comment = ""]) [*line 195*]

Function Parameters:

- *string* **\$path** the (absolute) pathname of the destination file
- *string* **\$comment** an optional comment to include in the ZIP-archive

open a file for subsequent output of ZIP-archive

this opens the file \$path for writing and also sets the zip_type to ZIP_TYPE_FILE. The optional \$comment is stored for future reference. The file must be closed afterwards via [CloseZip\(\)](#).

bool function Zip::OpenZipstream(\$name, [\$comment = ""]) [*line 223*]

Function Parameters:

- *string* **\$name** the name of the ZIP-archive that is suggested to the browser
- *string* **\$comment** an optional comment to include in the ZIP-archive

start with a stream (direct output) indicating an application/zip type of content

this starts the output of the ZIP-archive directly to the browser. The Content-Type and the Content-Disposition are set by sending headers. The stream must be closed afterwards via [CloseZip\(\)](#).

bool function Zip::zip_add_data(&\$data, \$filename, \$comment, \$timestamp) [*line 415*]

Function Parameters:

- *string* **&\$data** a pointer to a buffer with data to add to the ZIP-archive
- *string* **\$filename** the preferred name of the file in the ZIP-archive

- *string* **\$comment** an optional comment for this specific file (could be "")
- *int* **\$timestamp** the unix timestamp to associate with the file

workhorse function to add data to the current ZIP-archive

This actually adds the data to the current ZIP-archive.

Note that we try to make a wise decision about compressed data: the compressed data should be smaller than the uncompressed data. If not, we don't bother and simply store the data as-is.

We also try to keep the number of copies of the data down to a minimum by not copying the \$data but selecting between \$data and \$zdata only when we are really ready to write output the data.

- **TODO** should we handle the possibility of an additional 4 bytes for DICTID (RFC1950, reference [2])?
- **TODO** should we handle the option of a better compression level (eg. level 9) in gzcompress()? we could check to see if CMF equals 0x78 and FLG is either 0x01, 0x5E, 0x9C or 0xDA the latter 4 values might have an effect on general purpose bit flag bits 2 and 3. for now we'll just keep it simple, but there might be a little something to improve here.

bool function Zip::zip_add_directories(\$pathname, \$timestamp) [*line 519*]

Function Parameters:

- *string* **\$pathname** contains 0, 1 or more directories that lead to the file that needs to be added
- *int* **\$timestamp** unix timestamp to associate with the directory

workhorse function to add 0, 1 or more directories to the current ZIP-archive

this routine works from top to bottom through the specified path, adding directories to the archive. If a particular directory was already added before, it is not added again. This information is based on the existence of the corresponding key in the central_directory array.

void function Zip::zip_error(\$function, \$message) [*line 600*]

Function Parameters:

- *string* **\$function** name of the function/method where the error occurred
- *string* **\$message** the error message to add

add an error message to the list of error messages

Package wasinstall Procedural Elements

install.php

/program/install.php - the main entrypoint for website installation

This is one of the main entry points for Website@School. Other main entry points are /admin.php, /cron.php, /file.php and /index.php. There is also /program/manual.php. Main entry points all define the constant WASENTRY. This is used in various include()ed files to detect break-in attempts.

- **Package** wasinstall
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: install.php,v 1.30 2016/06/16 14:19:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **TODO** how prevent third party-access to install.php after initial install? .htaccess? !exists(..../config.php)?
- **TODO** we should make sure that autosession is disabled in php.ini, otherwise was won't work
- **TODO** we should make sure that we can actually set cookies (necessary when logging in).
- **TODO** we should make sure that register globals is off
- **License** [GNU AGPLv3+Additional Terms](#)

```
INSTALL_DIALOG_CANCELLED = 11 [line 52]
INSTALL_DIALOG_CMS = 4 [line 45]
INSTALL_DIALOG_COMPATIBILITY = 6 [line 47]
INSTALL_DIALOG_CONFIRM = 7 [line 48]
INSTALL_DIALOG_DATABASE = 3 [line 44]
INSTALL_DIALOG_DONE = 9 [line 50]
INSTALL_DIALOG_DOWNLOAD = 10 [line 51]
INSTALL_DIALOG_FINISH = 8 [line 49]
INSTALL_DIALOG_INSTALLTYPE = 1 [line 42]
INSTALL_DIALOG_LANGUAGE = 0 [line 41]
INSTALL_DIALOG_LICENSE = 2 [line 43]
```

```
INSTALL_DIALOG_USER = 5 [line 46]
PROJECT_SITE = websiteatschool.eu [line 53]
WASENTRY = __FILE__ [line 37]
```

Valid entry points define WASENTRY; prevents direct access to include()'s.

string function install_script_name(\$full_wasentry_path) [line 4051]

Function Parameters:

- *string* **\$full_wasentry_path** full path of the entry point, e.g.
'/home/httpd/htdocs/was/program/install.php'

determine the name of the executing script (the entry point)

this routine tries to reach consensus about the name of the script that was the entry point. This is not as easy as it sounds. Here are some real-world examples in three variations:

- `http://exemplum.eu/was/index.php`
- `http://exemplum.eu/was/index.php?area=1&node=23`
- `http://exemplum.eu/was/index.php/1/23/Welcome`

The object of the exercises in the examples below is to arrive at the value '/was/index.php' using the elements from the global array \$_SERVER. Note that the case for index.php has additional variations, e.g.

- `http://exemplum.eu/was`
- `http://exemplum.eu/was/?area=1&node=23`

Example 1 - a simple Linux-server and `http://exemplum.eu/was/index.php`

```
DOCUMENT_ROOT => /home/httpd/htdocs
SCRIPT_FILENAME => /home/httpd/htdocs/was/index.php
REQUEST_URI => /was/index.php
SCRIPT_NAME => /was/index.php
PHP_SELF => /was/index.php
PATH_INFO => (undefined)
```

Example 2 - a simple Linux-server and `http://exemplum.eu/was/index.php?area=1&node=23`

```
DOCUMENT_ROOT => /home/httpd/htdocs
SCRIPT_FILENAME => /home/httpd/htdocs/was/index.php
REQUEST_URI => /was/index.php?area=1&node=23
SCRIPT_NAME => /was/index.php
PHP_SELF => /was/index.php
PATH_INFO => (undefined)
```

Example 3 - a simple Linux-server and `http://exemplum.eu/was/index.php/1/23/Welcome`

```
DOCUMENT_ROOT => /home/httpd/htdocs
SCRIPT_FILENAME => /home/httpd/htdocs/was/index.php
REQUEST_URI => /was/index.php/1/23/Welcome
SCRIPT_NAME => /was/index.php
PHP_SELF => /was/index.php/1/23/Welcome
PATH_INFO => /1/23/Welcome
```

Example 4 - an ISP running php via CGI and `http://exemplum.eu/was/index.php`

```
DOCUMENT_ROOT => /usr/local/WWW/E/e/exemplum/htdocs
SCRIPT_FILENAME => /usr/local/WWW/E/e/exemplum/htdocs/cgi-bin/php
REQUEST_URI => /was/index.php
SCRIPT_NAME => /cgi-bin/php
PHP_SELF => /was/index.php
```

```
PATH_INFO      => /was/index.php
```

Example 5 - an ISP running php via CGI and <http://exemplum.eu/was/index.php?area=1&node=23>

```
DOCUMENT_ROOT => /usr/local/WWW/E/e/exemplum/htdocs
SCRIPT_FILENAME => /usr/local/WWW/E/e/exemplum/htdocs/cgi-bin/php
REQUEST_URI => /was/index.php?area=1&node=23
SCRIPT_NAME => /cgi-bin/php
PHP_SELF => /was/index.php
PATH_INFO => /was/index.php
```

Example 6 - an ISP running php via CGI and <http://exemplum.eu/was/index.php/1/23/Welcome>

Server reply: No input file specified.

Simply using `SCRIPT_NAME` as per PHP Documentation won't work (see examples 4 and 5). Simply using `PHP_SELF` is also problematic (see example 3) because it equates to user input. Another problem is the use of symbolic links. The ISP running php via CGI shows this value for `__FILE__` (in `index.php`):

```
__FILE__ => /usr/local/WWW/E/.5c1/e/exemplum/htdocs/was/index.php
```

so this simple assertion of the calculated value `'/was/index.php'` fails:

```
$DOCUMENT_ROOT.'/was/index.php' == __FILE__
```

because of the `/.5c1` within the `__FILE__` path. However, this might be solved by looking at the `realpath()` of the left hand side because that resolves the 'hidden' link within `$DOCUMENT_ROOT`.

All in all the parameter `REQUEST_URI` shows the most consistent information never mind the appended parameters like `node=23`, so we start there.

The strategy is as follows.

1. If `REQUEST_URI` begins with `SCRIPT_NAME` then `SCRIPT_NAME` is the answer (works for 1/2/3)
2. If `REQUEST_URI` begins with `PHP_SELF` then `PHP_SELF` is the answer (works for 4/5)
If these two attempts fail, we try a different approach. We can determine the filename of the script (via `basename($full_wasentry_path)`) and try to match that with either `SCRIPT_NAME` or `PHP_SELF`. Specifically we look at the case there the filename is omitted ie. <http://exemplum.eu/was/> instead of <http://exemplum.eu/was/index.php>. This yields a `REQUEST_URI` that fails in tests 1 and 2 above.
3. If `(basename(__FILE__) == basename(SCRIPT_NAME))` then `SCRIPT_NAME` is the likely answer (works for 1/2/3)
4. If `(basename(__FILE__) == basename(PHP_SELF))` then `PHP_SELF` is the likely answer (works for 4/5)

Finally, as a double check we assert that the `DOCUMENT_ROOT` together with the answer actually yields the `__FILE__` path (resolving symlinks along the way). If it doesn't I'd say there is something going terribly wrong, wrong enough to warrant an emergency exit.

In other words: if there is only a slight doubt about the correct answer we simply `die()`;

Note that an almost identical routine [wasentry_script_name\(\)](#) is used in the main program via [init.php](#).

```
include_once dirname(__FILE__)."/version.php" [line 59]
```

```
include_once dirname(__FILE__)."/lib/utf8lib.php" [line 60]
```

demodata.php

/program/install/demodata.php - code to install the main demodata

this file, included from /program/install.php, contains a single routine which installs the main demodata. This is done only once during a fresh install of a site.

- **Package** wasinstall
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: demodata.php,v 1.28 2016/06/16 13:48:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function demodata(&\$messages, &\$config) [line 91]

Function Parameters:

- **array &\$messages** used to return (error) messages to caller
- **array &\$config** pertinent information about the site and also returns additional demo data

insert basic demonstration data; the foundation for the module/theme demonstration data

this routine inserts all sorts of demonstration data as a foundation for the demonstration of various modules and themes.

The array &\$messages is used to pass (error) messages back to the caller. The overall result returned is TRUE on success, or FALSE on failure.

The parameter &\$config is used to communicate essential information about the site that is being installed, such as the main URL and the various directories. Also the information about the first user account is passed; this can be used to setup alerts etc. Finally, this array is used to return the three numbers of the three demonstration areas created.

The first demonstration area is a public area. This would be the area to show off all bells and whistles of the CMS. The second demonstration area is a private area. This area could be used to show off an intranet-type of application, maybe accessible only to members of the team. The third area is an in-active area, just for the heck of it.

Note that the demonstration data is to be translated. All translations can be found in `/program/install/languages/LL/demodata.php` where LL indicates the language code. The language to use is specified in the parameter `$config['language_key']`.

This routine is completely self-contained, even the translations are handled manually here.

As an added bonus for other demodata routines (eg. the installers of demo data for modules and themes) the main demodata strings are added to `$config` and also a set of handy search/replace pairs. After calling this routine, the `$config` array contains the following.

```
$config['language_key'] => install language code (eg. 'en')
$config['dir']          => path to CMS Root Directory (eg. /home/httpd/htdocs)
$config['www']         => URL of CMS Root Directory (eg. http://exemplum.eu)
$config['progdir']     => path to program directory (eg. /home/httpd/htdocs/program)
$config['progwww']     => URL of program directory (eg. http://exemplum.eu/program)
$config['datadir']     => path to data directory (eg. /home/httpd/wasdata/alb2c3d4e5f6)
$config['title']       => the name of the site
$config['user_username'] => userid of webmaster (eg. wblader)
$config['user_full_name'] => full name of webmaster (eg. Wilhelmina Bladergroen)
$config['user_email']  => email of webmaster (eg. w.bladergroen@exemplum.eu)
$config['user_id']     => numerical user_id (usually 1)
$config['demo_salt']   => password salt for all demodata accounts
$config['demo_password'] => password for all demodata accounts
$config['friendly_url'] => TRUE if friendly URLs are allowed (added 2013-07-03)
$config['demo_areas']  => array with demo area data
$config['demo_groups'] => array with demo group data
$config['demo_users']  => array with demo user data
$config['demo_nodes']  => array with demo node data
$config['demo_string'] => array with demo strings from /program/install/languages/LL/demodata.php
$config['demo_replace'] => array with search/replace pairs to 'jazz up' the demo strings
```

bool function `demodata_alerts(&$messages, &$config, &$tr)` [line 1129]

Function Parameters:

- **array &\$messages** used to return (error) messages to caller
- **array &\$config** pertinent information about the site
- **array &\$tr** translations of demodata texts

create a few alerts

bool function `demodata_areas(&$messages, &$config, &$tr)` [line 174]

Function Parameters:

- **array &\$messages** used to return (error) messages to caller
- **array &\$config** pertinent information about the site
- **array &\$tr** translations of demodata texts

create three areas + themes

bool function demodata_sections_pages(&\$messages, &\$config, &\$tr) [line 635]

Function Parameters:

- **array &\$messages** used to return (error) messages to caller
- **array &\$config** pertinent information about the site; receives copy of nodes array on return
- **array &\$tr** translations of demodata texts

create a few sections and pages

this constructs a complete public area with some pages and sections and also the 'frugal' theme is configured for this area. The information about the nodes (including the assigned node_id) is copied to \$config['demo_nodes'] for the caller's perusal.

bool function demodata_users_groups(&\$messages, &\$config, &\$tr) [line 334]

Function Parameters:

- **array &\$messages** used to return (error) messages to caller
- **array &\$config** pertinent information about the site, also receives copy of users/groups data
- **array &\$tr** translations of demodata texts

create a handful of users/groups/capacities/acls

This routine creates the following 4 groups:

- faculty (principals and teachers)
- team (principals and teachers and all other employees)
- seniors (pupils in grades 5 to 8)
- juniors (pupils in grades 1 to 4)

The following 7 group/capacities are also created

- faculty/principal (3)
- faculty/member (4)
- team/member (4)
- seniors/pupil (1)
- seniors/teacher (2)
- juniors/pupil (1)
- juniors/teacher (2)

The following 8 users are also created

- Amelia Cackle (acackl): Faculty/Principal, Team/Member

- Maria Montessori (mmonte): Faculty/Member, Team/Member, Seniors/Teacher
- Helen Parkhurst (hparkh): Faculty/Member, Team/Member, Juniors/Teacher
- Freddie Frinton (ffrint): Team/Member
- Andrew Reese (andrew): Seniors/Pupil
- Catherine Hayes (catherine): Seniors/Pupil
- Herbert Spencer (herberd): Juniors/Pupil
- Georgina King (georgina): Juniors/Pupil

Every user and every group/capacity gets their own acl

- faculty/principal: access to all private areas
- faculty/member: access to intranet in `$config['demo_areas']['private']['area_id']`
- others get no special privileges

The arrays with groups (including the assigned `group_id`) and users (with the assigned `user_id`) are stored in `$config['demo_groups']` and `$config['demo_users']`, for the caller's perusal.

- **TODO** get rid of the \$wizard kludge!
- **TODO** should we append an underscore to the userpaths to make sure we don't clash with the first user account?
- **TODO** should we also add `groups_capacities`, `acls`, `users_groups_capacities` to `$config` or are users and groups enough?

index.php

/program/install/index.php - redirector for website installation

The sole purpose of this file is to redirect users from /program/install to /program/install.php. The latter is the main entry point for website installation.

- **Package** wasinstall
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: index.php,v 1.7 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

demodata.php

/program/install/languages/en/demodata.php - translated messages for
/program/install/demodata.php (English)

This file holds the English texts that are used in the part of the installer that inserts the demonstration data. It is the basis for all other language files.

- **Package** wasinstall
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: demodata.php,v 1.16 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/en/install.php - translated messages for
/program/install.php (English)

This file holds the English texts that are used in the installer. It is the basis for all other language files.

- **Package** wasinstall
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: install.php,v 1.8 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

demodata.php

/program/install/languages/nl/demodata.php - translated messages for
/program/install/demodata.php (Dutch)

This file holds the Dutch texts that are used in the part of the installer that inserts the demonstration data.

- **Package** wasinstall
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: demodata.php,v 1.12 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/nl/install.php - translated messages for /program/install.php
(Dutch)

This file holds the Dutch translations that are used in the installer.

- **Package** wasinstall
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: install.php,v 1.8 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

tabledata.php

/program/install/tabledata.php defines core data in a generic way

This file contains the essential data for a new installation, i.e. the items in the configuration table that should exist, etc. This is all done in a generic (database-independent) way. This file defines an array called '\$tabledata' which contains one or more arrays with a tablename and yet another array with fieldname/fieldvalue pairs. This construction with nested arrays is converted to an actual SQL-statement in a function. That function also takes care of the table prefix, so we can simple refer to the bare tablenames here.

The reason to use this nested array construction is that it is easier to see which field gets which value compared to a (possibly very long) SQL-statement. Furthermore you don't need to worry about prefixing the table name and it is almost impossible to mismatch the number of fields and the number of values because they are combined in a \$key => \$value pair. Finally, all strings are automagically escaped with \$DB->escape() in the function that constructs the actual SQL-statement.

This definition file uses the PHP variable types, i.e. if you want to insert a number, you can specify a number (without quotes) and for a boolean you can use the PHP-values TRUE or FALSE. Here's an example. \$tabledata = array();

```
$tabledata[] = array('table' => 'tablename_without_prefix',  
    'fields' => array(  
        'string_field' => 'This is a string, even with unescaped "quotes"',  
        'boolean_field' => TRUE,  
        'integer_field' => 123,  
        'date_field' => '2008-02-01 23:34:45',  
        'double_field' => 1.234567,  
        'field_with_null_value' => NULL  
    )  
);
```

Note that a date field is handled like a string field.

- **Package** wasinstall
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: tabledata.php,v 1.14 2016/04/11 12:09:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

tabledefs.php

/program/install/tabledefs.php defines all core tables in a generic way

This is the main data definition for Website@School. This file is used by the installation script [install.php](#) to create all main tables.

Here is a reminder for the allowed parameters for field- and key definitions. FIELDS | name | type | len | dec | unsigned* | notnull | default | enum_values | comment |

-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										
serial*	req	req	-	-	-	-	-	-	-	opt
bool	req	req	-	-	-	opt	opt	-	-	opt
short	req	req	opt	-	opt	opt	opt	-	-	opt
int	req	req	opt	-	opt	opt	opt	-	-	opt
long	req	req	opt	-	opt	opt	opt	-	-	opt
float	req	req	opt	opt	opt	opt	opt	-	-	opt
double	req	req	opt	opt	opt	opt	opt	-	-	opt
decimal	req	req	opt	opt	opt	opt	opt	-	-	opt
number	req	req	opt	opt	opt	opt	opt	-	-	opt
varchar	req	req	opt	-	-	opt	opt	-	-	opt
enum	req	req	opt	-	-	opt	opt	req	-	opt
char	req	req	opt	-	-	opt	opt	-	-	opt
text	req	req	-	-	-	opt	-	-	-	opt
longtext	req	req	-	-	-	opt	-	-	-	opt
blob	req	req	-	-	-	opt	-	-	-	opt
longblob	req	req	-	-	-	opt	-	-	-	opt
date	req	req	-	-	-	opt	opt	-	-	opt
time	req	req	-	-	-	opt	opt	-	-	opt
datetime	req	req	-	-	-	opt	opt	-	-	opt
timestamp	req	req	-	-	-	opt	opt	-	-	opt

INDICES | name | type | unique | fields | reftable | reffields | comment |

-----+-----+-----+-----+-----+-----+-----+						
primary	-	req	-	req	-	opt
index	opt	req	opt	req	-	opt
foreign	opt	req	-	req	req	opt

req = required, opt = optional, - = not allowed

- **Package** wasinstall
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: tabledefs.php,v 1.14 2016/05/18 05:56:38 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker

- **TODO** automatically create appropriate sequence name for serial fields??? or add seqdefs too?
- **License** [GNU AGPLv3+Additional Terms](#)

Package wasinstall Classes

Class InstallWizard

[line 112]

class for performing installation tasks

Overview

Dialog screens

The installer basically consists of some six dialog screens where the user is supposed to enter some data (e.g. language, install type, etc.). Each of those screens has a [Next] button and most screens have a [Previous] button. Also, every screen has a [Cancel] button. The [Cancel] button always immediately leads to the Cancel screen and the whole process is stopped (by resetting the collected information in `$_SESSION['INSTALL']`). The [Next] button always validates/processes the data the user entered. If the results are good, we go to the next step. If the processing fails, we stay where we are (ie. the current dialog is re-displayed). The [Previous] button backs up one step, without saving or storing the user entered data; the user **MUST** press [Next] to save the data entered.

The Finish-dialog

The FINISH-dialog has no [Previous] button, because all the real work is done when the user presses [Next] in the CONFIRM-dialog. This is a one-time action (creating tables, filling with demodata, etc.) so it makes no sense to backup at that point.

The stage variable and backing up via the menu

The variable 'stage' moves along with the highest dialog the user has successfully reached. This variable is responsible for greying out/disabling the menu options. The menu can be used to jump back a few steps in the procedure. However, once the transition from CONFIRM to FINISH is made, it is no longer possible to return to previous steps (it makes no sense to do so because at that point the real work is already done). The jump to a particular step/dialog is done via the GET-parameter 'step'. The buttons all work via the POST'ed parameter dialog.

Special cases

There are a few special cases:

- download: this yields an immediate download of the constructed config.php and no further dialog is displayed
- done: this is a pseudo-dialog. In effect it is a redirect to the newly created site (either admin.php or index.php or perhaps manual.php).

- **Package** wasinstall

InstallWizard::\$license

string = [line 120]

- **Var** \$license ready-to-use HTML-code with the text of the license from /program/license.html

InstallWizard::\$messages

string = array() [line 114]

- **Var** \$messages collects error messages if any

InstallWizard::\$results

array = array() [line 117]

- **Var** \$results collects outcome of various compatibility results in human readable form

InstallWizard::\$time_start

string = [line 123]

- **Var** \$time_start keeps track of execution time of this script

Constructor *void* function InstallWizard::InstallWizard() [*line 138*]

constructor

this constructs the install wizard and also makes sure that the INSTALL-array (kept in the \$_SESSION array) is initialised with default values if it did not already exist.

Also we check if the user wants to set or change the debug value. If set to non-0, we show extra information later on (and also set the parameter in \$CFG in config.php).

Finally, we do record the microtime as early as possible, to calculate elapsed time after we are all done.

string function InstallWizard::appropriate_legal_notices([\$high_visibility = FALSE], [\$m = ""]) [*line 2543*]

Function Parameters:

- *bool* **\$high_visibility** if TRUE we return a text-only link, otherwise a clickable image
- *string* **\$m** margin to improve readability of generated code

construct a link to appropriate legal notices as per AGPLv3 section 5

This routine constructs ready-to-use HTML-code for a link to the Appropriate Legal Notices, which are to be found in /program/about.html. Depending on the highvisibility flag we either generate a text-based link or a clickable image.

The actual text / image to use depends on the global constant WAS_ORIGINAL. This constant is defined in /program/version.php and it should be TRUE for the original version of Website@School and FALSE for modified versions.

In the former case the anchor looks like 'Powered by Website@School', in the latter case it will look like 'Based on Website@School', which is in line with the requirements from the license agreement for Website@School, see /program/license.html.

IMPORTANT NOTE

Please respect the license agreement and change the definition of WAS_ORIGINAL to FALSE if you modify this program (see /program/version.php). You also should change the file '/program/about.html' and add a 'prominent notice' of your modifications.

Note: a comparable routine can be found in [waslib.php](#).

string function InstallWizard::button(\$button) [line 2821]

Function Parameters:

- *string* **\$button** indicates which button to create, e.g. 'next', 'previous', 'cancel', 'finish', 'ok'.

shorthand for creating a submit button in the correct style

bool function InstallWizard::check_compatibility() [line 1317]

check certain compatibility issues and optionally return test results

this routine performs a few tests and returns an overall go/nogo signal Human readable test results are stored in \$this->results. Return TRUE on passing all tests, FALSE otherwise + errors in \$this->messages

- **TODO** add more tests, e.g. for gd, safe_mode, memory limit, etc.

bool function InstallWizard::check_for_nameclash(\$demodata, \$label, \$username) [line 3251]

Function Parameters:

- *bool* **\$demodata** is installation of demodata requested?
- *string* **\$label** name of the username field in dialog
- *string* **\$username** the proposed username for the webmaster account

check for name clash of new user (webmaster) and user accounts from demodata

This routine checks to see if the name the webmaster supplied is not one of the demodata accounts. If so, we flag the error in the messages.

Note: The list of accounts must be updated whenever the demodata is updated. This is a kludge but I'll leave it this way for the time being. See also the main file [demodata.php](#).

bool function InstallWizard::check_license() [line 506]

check if the user accepts the licences

This is the companion routine for [show_dialog_license\(\)](#). It checks whether the user dutifully typed 'I agree'. Note that we check caseInsensitive, so 'i agree' is acceptable, as is 'I aGrEe'. Also note that we accept non-ascii UTF-8, say "J'accept'ee" or "J'ACCEPT'EE" which has an e-acute. Case folding to lowercase accepts all those variants.

bool function InstallWizard::check_validation() [line 1568]

shorthand to check the validation status of the relevant dialogs

this checks the various validation flags. If a flag is false, the corresponding error message is added to \$this->messages and the function returns FALSE.

void function InstallWizard::clamscan_installed(&\$clamscan_path, &\$clamscan_version) [line 3534]

Function Parameters:

- *string &\$clamscan_path* path to binary program (output)
- *string &\$clamscan_version* version of the program we found (output)

try to locate clamdscan or clamscan on the server

This routine checks to see if either clamdscan or clamscan can be found somewhere. This is done via educated guessing. On success we return the path to the binary program file in \$clamscan_path and the output of the version command in \$clamscan_version. The function returns TRUE if we did find the clamscan program.

Note that we scan the directories with opendir/readdir/closedir rather than rely on file_exists() etc. because we would not find binaries with permissions 0711, which would otherwise be perfectly acceptable to execute with exec().

Also note that we suppress PHP-error messages that might be visible when we tread outside the open_basedir.

string function InstallWizard::construct_config_php() [line 2330]

prepare a configuration file based on the collected information

bool function InstallWizard::create_tables(\$filename) [line 3393]

Function Parameters:

- *string \$filename* contains the table definitions

create tables in database via include()'ing a file with tabledefs

- **Uses \$DB**

double function InstallWizard::diff_microtime(\$time_start, \$time_stop) [line 3913]

Function Parameters:

- *string* **\$time_start** starting time as a string (fractional seconds, space, seconds)
- *string* **\$time_stop** ending time as a string (fractional seconds, space, seconds)

Calculate the difference between two microtimes (borrowed from init.php)

void function InstallWizard::end_session_and_redirect() [line 2160]

unset installation data, end session and redirect the user to elsewhere

this redirects the user to one of the possible destinations as selected in the finish dialog (see [show_dialog_finish\(\)](#)). Also, the INSTALL-array is unset, effectively erasing the collected data from the session. Subsequently the session cookie is reset, effectively ending the session. The effect is that the user has to start over if she returns to install.php. (The equivalent of 'logging off').

void function InstallWizard::errorcount_bump() [line 3096]

increment the error counter and perhaps slow things down

this routine counts the number of errors. It is used to count the number of attempts to guess a valid host/username/passord triplet. If the number of errors reaches 3, a delay is added (the installation program sleeps for a while). On every additional error an extra delay is added (3 seconds at a time) until the total delay reaches 24 seconds. At that point the collected data are reset and effectively the user has to start over.

void function InstallWizard::errorcount_reset() [line 3113]

reset the error counter

this routine resets the effect of [errorcount_bump\(\)](#).

int function InstallWizard::fetch_license(&\$license) [line 2879]

Function Parameters:

- **string &\$license** receives ready-to-use HTML-code with the text of the license from /program/license.html

helper to retrieve the text of the LICENSE AGREEMENT for Website

bool function InstallWizard::gd_supported(&\$details) [line 3641]

Function Parameters:

- **string &\$details** returns detailed information about GD version and supported file formats

retrieve information about GD and supported graphics file formats

this routine determines whether GD is enabled and which graphics file formats are supported (check for GIF, JPG and PNG). If GD is not installed or if none of these three formats are supported, the routine returns FALSE. Details (version number, supported formats) are returned in &\$details.

GIF is a special case due to patent issues: there is a distinction between read support and write support (see the PHP-documentation for details). GD-version >= 2.0.28 provides full support. This routine makes the distinction between fully supported (GIF: Yes), reading but not writing (GIF: Readonly) and not supported (GIF: No).

There is an issue with the return value of gd_info(). Here are two real world examples:

```
PHP 4.3.6 (Linux): gdinfo = Array ( [GD Version] => bundled (2.0.22 compatible)
[FreeType Support] => 1 [FreeType Linkage] => with freetype [T1Lib Support] =>
[GIF Read Support] => 1 [GIF Create Support] => [JPG Support] => 1
[PNG Support] => 1 [WBMP Support] => 1 [XBM Support] => 1 [JIS-mapped
Japanese Font Support] => )
```

```
PHP 5.3.5 (XAMPP): gdinfo = Array ( [GD Version] => bundled (2.0.34 compatible)
[FreeType Support] => 1 [FreeType Linkage] => with freetype [T1Lib Support] =>
[GIF Read Support] => 1 [GIF Create Support] => 1 [JPEG Support] => 1
[PNG Support] => 1 [WBMP Support] => 1 [XPM Support] => [XBM Support]
=> 1 [JIS-mapped Japanese Font Support] => )
```

Note the subtle difference between 'JPG Support' and 'JPEG Support'. Checking for the first one of those yields a PHP Notice: "Undefined index: JPG Support (...)" in the second installation. *sigh* We work around it with isset(), on `_all_` elements of gdinfo() to be certain.

array function InstallWizard::get_default_install_values() [line 2200]

return an array with default configuration values

this routine tries to calculate/guess the best default values for config.php. We do so by looking in the global `$_SERVER` variable. If that doesn't work, we simply make up sensible values. In the end it is up to the user to enter the correct data; the values here are mere defaults.

A first-time install should be possible without changing/editing the default values, i.e. a standard Next-Next-Finish-type of installation.

- **TODO** should we check the program version versus the stored program version here?
- **TODO** there is something wrong with the default for `$cms_www`; **FIXME** (commented out for now)

array function InstallWizard::get_dialogdef_cms() [line 1018]

fill an array with necessary information for the cms dialog

Note that this is a very light-weight implementation of the dialogdef idea used in the main program: we don't do fancy stuff with labels, hotkeys, etc. KISS, because I don't want to rely on all the libraries of the main program with all their interconnections and dependencies; the installer should more or less be a stand-alone application.

Note: the order of 'cms_demodata' and 'cms_demodata_password' is important: the field 'cms_demodata' must come first. If not, the validation of the password is not skipped if demodata is left unchecked by the user. See also [save cms\(\)](#).

array function InstallWizard::get_dialogdef_database() [line 728]

fill an array with necessary information for the database dialog

Note that this is a very light-weight implementation of the dialogdef idea used in the main program: we don't do fancy stuff with labels, hotkeys, etc. KISS, because I don't want to rely on all the libraries of the main program with all their interconnections and dependencies; the installer should more or less be a stand-alone application.

array function InstallWizard::get_dialogdef_finish() [line 2094]

fill an array with necessary information for finish / jump dialog

Note that this is a very light-weight implementation of the dialogdef idea used in the main program: we don't do fancy stuff with labels, hotkeys, etc. KISS, because I don't want to rely on all the libraries of the main program with all their interconnections and dependencies; the installer should more or less be a stand-alone application.

array function InstallWizard::get_dialogdef_installtype() [line 417]

fill an array with necessary information for installtype dialog

Note that this is a very light-weight implementation of the dialogdef idea used in the main program: we don't do fancy stuff with labels, hotkeys, etc. KISS, because I don't want to rely on all the libraries of the main program with all their interconnections and dependencies; the installer should more or less be a stand-alone application.

array function InstallWizard::get_dialogdef_language() [line 328]

fill an array with necessary information for language dialog

Note that this is a very light-weight implementation of the dialogdef idea used in the main program: we don't do fancy stuff with labels, hotkeys, etc. KISS, because I don't want to rely on all the libraries of the main program with all their interconnections and dependencies; the installer should more or less be a stand-alone application.

array function InstallWizard::get_dialogdef_user() [line 1212]

fill an array with necessary information for the first user dialog

Note that this is a very light-weight implementation of the dialogdef idea used in the main program: we don't do fancy stuff with labels, hotkeys, etc. KISS, because I don't want to rely on all the libraries of the main program with all their interconnections and dependencies; the installer should more or less be a stand-alone application.

array function InstallWizard::get_list_of_install_languages() [line 2845]

retrieve a list of available languages by querying the file system for install.php translation files

this routine constructs a list of language codes and language names (in the languages themselves) based on the language subdirectories available under /program/install/. The resulting array of code-name-pairs is sorted by name.

Note that because the names of the languages are expressed in the languages themselves, this routine has the side-effect of reading all of the available language files into memory (see [t\(\)](#)).

Also note that the sort order is based on UTF-8 byte sequences, i.e. the order is 7-bit ASCII bytes, followed by 2-byte, 3-byte and 4-byte UTF-8 sequences. No problem for this list of languages IMHO.

array function InstallWizard::get_manifests(\$path) [line 3366]

Function Parameters:

- *string* **\$path** top directory for the search for manifest files

retrieve an array of manifests for modules, themes or languages

this examines the file system starting in the directory \$path, looking for manifest files. These manifest files are named after the subdirectory they are in as follows. Example: If \$path is /program/modules, this routine steps through that directory and may find subdirectories 'htmlpage', 'guestbook' and 'forum'. Eventually these manifest files are include()'d:

```

                                /program/modules/htmlpage/htmlpage_manifest.php,
/program/modules/guestbook/guestbook_manifest.php
/program/modules/forum/forum_manifest.php.
```

Every manifest file must describe the module (or language or theme) via the following construct:

```
$manifests['htmlpage'] = array('name' => 'htmlpage', ..., 'cron_interval' => 0);
```

After processing all the subdirectories of \$path, the resulting array \$manifests is returned. Note that pseudo-directories like '.' and '..' are not considered. Also, subdirectories 'foo' without the file 'foo_manifest.php' are also ignored.

Note that the name of the manifest file itself is also stored in the array, but excluding the subdirectory name.

string function InstallWizard::get_menu(\$dialog, \$stage, [\$m = ""]) [line 2602]

Function Parameters:

- *int* **\$dialog** indicates the current dialog
- *int* **\$stage** indicates the highest numbered dialog that was already reached
- *string* **\$m** margin for better readability of generated HTML-code

construct a clickable menu which helps the user to jump back and forth in the funnel

this constructs a menu that allows the user to jump to a another step in the procedure when appropriate. Two parameters are important: the \$dialog and the the \$stage. The \$dialog indicates the current dialog, i.e. the dialog that is currently displayed in the content area. This item in the menu is emphasised (e.g. the link is underlined via the style sheet). The \$stage indicates how far we are in the procedure. The installation consists of some eight steps, and the used is encouraged to perform all steps in the natural order, by repeatedly pressing the [Next] button in the dialogs. Every time a dialog appears to have valid data, the \$stage is incremented.

All the menu items after the current stage are greyed out and basically inaccessible for the user. Menu items before the current stage are accessible, so it is possible to jump backwards to dialogs that were already processed but the only way to advance to a new screen is to use the [Next] (and provide valid data, obviously).

The greyed-out menu items have a href property consisting of a simple "#". This is interpreted by the browser as a relative link within the current page. The effect is that the current page stays on the screen, including any unsaved data the user may already have entered. By showing the links in a different colour (grey and not blue), the user can visually see which items are clickable and which are not.

By showing all the menu items and greying-out the inaccessible ones we effectively build a funnel and at the same time indicating which steps will follow in the procedure.

There is a special case when \$stage hits the FINISH-dialog. If the user has not yet reached that stage, all dialogs before the finish-dialog are active and the rest is greyed-out. Once the \$stage reaches the FINISH-dialog, all previous dialogs become instantly unreachable. This is because the step between the confirmation dialog and the finish dialog is the place where the actual installation takes place. That is a one-time operation and the user should not be able to jump backwards and change data after all the actual installation work is already done.

Note that the menu item to download the config.php is displayed before the finish dialog. This is because it appears more logical to me (but YMMV).

array function InstallWizard::get_options_db_type() [line 3062]

construct a list of database options

This constructs an array with key-value-pairs indicating all available databases. Currently (february 2015) there is only one database supported: MySQL. However, for backwards compatibility both the 'mysqli' and the deprecated 'mysql' APIs are supported. In the future we may support more databases, such as PostgreSQL.

Note that we present ordinary users with a single choice (MySQL) while distinguishing between the two variants. Anyone using the debug-option will have an exact choice between the two flavours.

string function InstallWizard::get_page([\$dialog_title = "], [\$menu = "], [\$content = "], [\$help_topic = 'install']) [line 2391]

Function Parameters:

- *string \$dialog_title* text to show in the browser's title bar (indicating where we are)
- *string \$menu* ready-to-use HTML-code comprising the menu at the left hand side of the page
- *string \$content* ready-to-user HTML-code holding the actual contents of the page
- *string \$help_topic* the topic or subtopic in the manual to link to

construct a complete HTML-page that can be sent to the user's browser

this routine constructs a string containing the complete page to send to the user's

browser, starting with the <html> opening tag and ending with the </html> closing tag. The constructed page is returned as a string.

This routine also peeks into the INSTALL-array, e.g. for the language key and the high_visibility flag.

If \$this->messages is not empty, the items in that array are displayed between the page header (logo+helpbutton) and the menu/content area. This is the feedback of the previous action to the user (if any).

Note that we hard-code the content type to be "text/html; charset=UTF-8" in a http-equiv right at the top of the <head>. This indicates the correct charset to the browser even if a generated page is stored by the user and later retrieved in which case there will be no http-header. (We cannot be sure that the user's browser correctly identifies the stored document as UTF-8 if that is not the default).

- **TODO** should we promote language and high_visibility to function parameters instead of using \$_SESSION directly?

mixed function InstallWizard::get_utf8_parameter_string(&\$gpc, \$name, [\$default_value = "], [\$error_value = "], [\$maximum_length = 255]) *[line 2950]*

Function Parameters:

- *array* **&\$gpc** a reference to the array to search (usually \$_POST, \$_GET or \$_COOKIE)
- *string* **\$name** the name of the variable to retrieve
- *mixed* **\$default_value** the value to return if parameter is not in \$gpc
- *mixed* **\$error_value** the value to return if parameter is not valid UTF-8
- *mixed* **\$maximum_length**

return a valid (unquoted) UTF-8 string parameter typically from \$_POST, or default value if none

this retrieves a parameter string, typically from \$_POST, and checks basic properties.

1. the string should not contain more than \$maximum_length characters
2. the string should be valid UTF-8

If it is not a valid UTF-8 string or the string is too long, \$error_value (default: empty string) is returned. If no string by the name \$name is present in \$gpc, the default value

`$default_value` is returned. If there WAS a valid UTF-8 string, it is automagically UNquoted in the process.

Note that we check the stringlength in two steps. First we check the worst-case UTF-8 string with 4-byte sequences for a length of $4 * \$maximum_length$. Then, after establishing that the string is valid UTF-8 we make a more precise check of the length expressed in chars rather than bytes.

array function InstallWizard::guess_url() [line 2694]

educated guesses for scheme, host and portname from `$_SERVER`

this routine tries to guess the various components of the url that was used to reach this script, based on the information in the global array `$_SERVER`. If no information can be guessed at all, the result is something like 'http://localhost'.

Note that the 'authority' is a combination of hostname and portnumber, but only if the portnumber is non-standard. For http and port 80, and https and port 443 the portnumber is suppressed, because these are the default ports for those schemes.

Note that we actually guess the url that should correspond with the document root.

bool function InstallWizard::insert_tabledata(\$filename) [line 3425]

Function Parameters:

- *string* **\$filename** contains the table definitions

fill tables in database via `include()`'ing a file with tabledata

- **Uses \$DB**

string function InstallWizard::invisible_test_image([\$m = ""]) [line 3900]

Function Parameters:

- *string* **\$m** margin for better readability of generated HTML-code

create a link to an invisible image to test the friendly URL feature

this routine is part of a trick to discover whether the web server understands a friendly url like /index.php/1/23/Welcome rather than /index.php?area=1&node=23. This trick works as follows.

1. By default we assume that the webserver does NOT understand friendly url's by setting the corresponding install parameter to FALSE (see [get_default_install_values\(\)](#))
2. In the 5th step (after the Database dialog, ie. 'hidden' behind a password) we embed an invisible image (see below) in the dialog.
3. The browser will attempt to retrieve the image and to (not) show it.
This means that we call \$WAS_SCRIPT_NAME with a friendly url appears to point to 1x1.gif.
4a. IF (and only if) the webserver understands this friendly URL, we arrive in [run\(\)](#) with _SERVER['PATH_INFO'] set to /friendly/test/1x1.gif. This gives us a chance to flip the parameter 'friendly_url' in the installation parameters from the default FALSE to TRUE 4b. If the webserver does NOT understand the friendly url, the image file will never be found, and we will never return the 1x1.gif and also never flip the setting from FALSE to TRUE. 5. Done

I chose the Website dialog for this trick, because you can only get there after you have entered a valid username/password for the database. This means that only genuine users will perform this trick; outsiders never get to that dialog without a valid password.

Obviously the setting is eventually entered in the configuration table, just like the title and the website_from_address etc.

Note: both the Install Wizard itself and the code that generates the test image (in [run\(\)](#)) access the session variables. According to the PHP documentation this works because the file that holds the session information is locked between the calls to session_start() and session_write_close(). The only visible effect might be that the one call has to wait for the other to finish. Oh well, it is only a 1x1 gif of 35 bytes. It probably will go unnoticed.

- Used by [InstallWizard::show_dialog_cms\(\)](#)
- Used by [InstallWizard::run\(\)](#)

void function InstallWizard::is_already_installed() [line 3271]

check for previous install

this routine checks to see if another installation should be allowed. Returns TRUE if the program was already installed or FALSE otherwise.

string function InstallWizard::magic_unquote(\$value) [*line 2916*]

Function Parameters:

- *string* **\$value** a string value that is conditionally unescaped

this circumvents the 'magic' in magic_quotes_gpc() by conditionally stripping slashes

This routine borrowed from waslib.php.

string function InstallWizard::microtime() [*line 3928*]

a trick for systems without the microtime function

if the system function does not exist, we return the current time with a 1-second resolution.

int|bool function InstallWizard::mysql_close(\$db_type, \$db_link) [*line 3855*]

Function Parameters:

- *string* **\$db_type** is either mysql or mysqli
- *string* **\$db_link**

unified mysql/mysqli wrapper for close

resource|object|bool function InstallWizard::mysql_connect(\$db_type, \$server, \$username, \$password) [*line 3735*]

Function Parameters:

- *string* **\$db_type** is either mysql or mysqli
- *string* **\$server** with optional port number eg. "dbserver:3306"
- *string* **\$username**
- *string* **\$password**

unified mysql/mysqli wrapper for connect

string|bool function InstallWizard::mysql_get_server_info(\$db_type, \$db_link) [*line 3755*]

Function Parameters:

- *string* **\$db_type** is either mysql or mysqli
- *string* **\$db_link**

unified mysql/mysqli wrapper for get_server_info

int|bool function InstallWizard::mysql_num_rows(\$db_type, \$result) [*line 3839*]

Function Parameters:

- *string* **\$db_type** is either mysql or mysqli
- *string* **\$result**

unified mysql/mysqli wrapper for num_rows

resource|object|bool function InstallWizard::mysql_query(\$db_type, \$db_link, \$query) [*line 3772*]

Function Parameters:

- *string* **\$db_type** is either mysql or mysqli
- *string* **\$db_link**
- *string* **\$query**

unified mysql/mysqli wrapper for query

string|bool function InstallWizard::mysql_real_escape_string(\$db_type, \$db_link, \$str) [*line 3806*]

Function Parameters:

- *string* **\$db_type** is either mysql or mysqli
- *string* **\$db_link**
- *string* **\$str** unescaped string

unified mysql/mysqli wrapper for real_escape_string

string|bool function InstallWizard::mysql_select_db(\$db_type, \$db_link, \$db_name) [*line 3789*]

Function Parameters:

- *string* **\$db_type** is either mysql or mysqli
- *string* **\$db_link**
- *string* **\$db_name**

unified mysql/mysqli wrapper for select_db

string|bool function InstallWizard::mysql_set_charset(\$db_type, \$db_link, \$charset) [*line 3823*]

Function Parameters:

- *string* **\$db_type** is either mysql or mysqli
- *string* **\$db_link**
- *string* **\$charset**

unified mysql/mysqli wrapper for set_charset

string function InstallWizard::mysql_utf8mb3(\$utf8str) [*line 3718*]

Function Parameters:

- *string* **\$utf8str** valid UTF-8 encoded string

message string to contain only 3-byte UTF8-sequences

this replaces an otherwise perfectly valid 4-byte UTF-8 sequence in \$utf8str with a 3-byte UTF-8 sequence equivalent with the Unicode replacement character U+FFFD.

The effect is that it leaves a hint that there used to be some character instead of silently discarding 4-byte sequences which MySQL does.

int|bool function InstallWizard::mysql_utf8_support(\$db_type, \$db_link) [*line 3688*]

Function Parameters:

- *string* **\$db_type** is either mysql or mysqli

- *resource* **\$db_link** the MySQL connection

determine the level of UTF-8 support based on MySQL-server version

MySQL support for UTF-8 was non-existent before 4.1.x and limited until 5.5.3. In this context 'limited' means: only the Basic Multilingual Plane (U+0000 ... U+FFFF) is supported, i.e. a maximum of 3-byte sequences per character.

As of 5.5.3 the full UTF-8 specification according to RFC 3629 is implemented. MySQL now has 'invented' yet another proprietary name for this character set: 'utf8mb4' (WTF?), and introduces the alias 'utf8mb3' for the pre 5.5.3 limited support for 'utf8' (WTF??), hinting that the meaning of 'utf8' may change in future versions to indicate 'utf8mb4' (WTF???). IM(NS)HO this is yet another reason to go looking for a decent replacement for MySQL. YMMV.

This routine returns exactly one of the values below (based on the server version).

- 0: there is no UTF-8 support available in this server
- 3: the limited 3-byte sequences are supported
- 4: full support for 4-byte sequences available, but using the stupid ad-hoc name 'utf8mb4' or the value FALSE if version information could not be obtained.

bool function InstallWizard::perform_installation() [*line 1628*]

perform the actual initialisation of the cms

this routine initialises the database: creates tables, inserts essential data (first user account, other defaults) and optional demonstration data.

The strategy is as follows.

- (1) manufacture a database object in the global \$DB
- (2A) create the main tables (from /program/install/tabledefs.php)
- (2B) insert essential data (from /program/install/tabledata.php)
- (2C) store the collected data (website title, etc.),
- (3) if necessary, create the data directory
- (4) record the currently available languages in the database
- (5) create the first useraccount,

Once the main part is done, install modules and themes based on the relevant information that is stored in the corresponding manifest-file by performing the following steps for each module and theme:

- (6A) insert a record in the appropriate table with active = FALSE
- (6B) create the tables (if any tables are necessary according to the manifest)
- (6C) install the item by including a file and executing function <item>_install()
- (6D) flip the active flag in the record from step 5A to indicate success

Subsequently the optional demodata is installed.

- (7A) a foundation is created via the function `demodata()` from `/program/install/demodata.php`
 - (7B) all modules + themes can add to the demo data via the appropriate subroutines
If all goes well, this routine ends with an attempt to
 - (8) save the `config.php` file at the correct location.
(it is not an error if that does not work; it only means that the user has to upload the `config.php` file manually.
-
- **TODO** should we save the `config.php` to the `datadir` if the main dir fails? Mmmm.... security implications?
 - **TODO** this routine badly needs refactoring

void function `InstallWizard::quasi_random_string($length, [$candidates = 36])` [line 3463]

Function Parameters:

- *int* **\$length** length of the string to generate
- *int* **\$candidates** number of candidate-characters to choose from

generate a string with quasi-random characters

This routine borrowed from [waslib.php](#).

Note that this too is an ASCII-centric routine: we only use plain ASCII letters and digits and nothing of the 64000 other Unicode characters in the Basic Multilingual Plane. The reason is simple: 7-bit ASCII characters have the best chance of getting through communication channels unmangled so there.

string function `InstallWizard::render_dialog($dialogdef, $m)` [line 2985]

Function Parameters:

- *array* **\$dialogdef** contains labels, values and other information describing input elements
- *string* **\$m** improves readability of generated code

quick and dirty dialogdef renderer

This is a small routine to render simple dialogs with strings, lists and checkboxes.

Note that every element in the \$dialogdef is in itself an array. Recognised elements in those arrays are:

- label (string): displayed before the actual input element
- help (string): additional information for the user, rendered after/under the input element
- value (mixed): the current value of the input element
- show (bool): if TRUE, the element is displayed/rendered, otherwise it is simply skipped
- type (enum): 's'=>string (text), 'p'=>password, 'l'=>list, 'b'=>bool(checkbox)
- options(array): array with key-value-pairs with acceptable choices (used in 'l' (list) elements)
- minlength(int): minimum length of an input string or password
- maxlength(int): maximum length of an input string or password

The names of the input elements are copied from the keys used in \$dialogdef.

void function InstallWizard::run() [*line 161*]

main dispatcher for the Installation Wizard

This routine termines what needs to be done and does it by calling the corresponding workhorse routines.

Note the special trick with the test for a friendly URL; see [invisible_test_image\(\)](#) for the gory details.

- Uses [InstallWizard::invisible_test_image\(\)](#)

string function InstallWizard::sanitise_filename(\$filename) [*line 3487*]

Function Parameters:

- *string* **\$filename** the string to sanitise

sanitise a string to make it acceptable as a filename/directoryname

This routine more or less borrowed from [waslib.php](#).

Note that this routine too is very ASCII-centric: in the end only ASCII-characters (52 letters, 10 digits and dash, dot and underscore) are allowed in the resulting file/directory name. However, by first mapping UTF-8 to ASCII (getting rid of diacriticals) we can make names

more readable.

bool function InstallWizard::save_cms() [line 852]

validate and store the CMS-data the user supplied

This is the companion routine for [show_dialog_cms\(\)](#). It stores the user-supplied data about the website (paths etc.) We always store the data in the global `_SESSION`, even if something goes wrong. This makes that the user will use the latest values when re-doing the dialog.

We try to validate the specified directories: they should at least exist. The datadirectory should also be writable by us. If it not exists we try to create it (and remove it after the test). Note that the PHP `safe_mode` may complicate things here.

- **TODO** also take `safe_mode` into account? Should that be a requirement for succesfull installation?

bool function InstallWizard::save_database() [line 594]

validate database information

This is the companion routine for [show_dialog_database\(\)](#). It stores the user-supplied data about the database. We always store the data in the global `_SESSION`, even if something goes wrong. This makes that the user will use the latest values when re-doing the dialog. However, only when the values are valid, the parameter `db_validated` is set to `TRUE`. This is used lateron (in the confirmation phase).

This routine doubles as a gate keeper. Every time the user makes a mistake, an error counter is incremented and the script pauses for some time (see [errorcount_bump\(\)](#)). If there are too many errors, the script resets the data collected sofar and the procedure starts from scratch. This is a (probably futile) attempt to make it harder to brute force an entry by repeatedly probing for database credentials. Unfortunately there is no easy way (at least one I can think of) to protect this script from repeated break-in attempts other than by simply removing or renaming this script. Oh well.

The following tests are performed:

- fields should not fail basic tests (min/max stringlength etc.)
- the database is not supposed to be in the list of 'forbidden' databases (e.g. 'test' or 'mysql')
- prefix must only use ASCII-letters A-Z or a-z, digits 0-9 or an underscore
- prefix must start with an ASCII-letter

If the above conditions are not satisfied we bail out immediately, without testing other information. Otherwise, we also perform these tests:

- is it possible to connect to the database server
- is it possible to select the specified database
- are there no tables in the database that start with the prefix

If something needs to be done about the prefix and we are in standard mode, we automatically switch to custom mode, allowing the user to edit the prefix too.

bool function InstallWizard::save_installtype() [line 384]

store the selected install type + high visibility flag

This is the companion routine for [show_dialog_installtype\(\)](#). It stores the user-supplied choices for custom install and high visibility. The values in the radio button should be 0 for standard and 1 for custom install. The value we keep is a boolean indicating a custom install yes or no.

bool function InstallWizard::save_language() [line 298]

store the selected language

This is the companion routine for [show_dialog_language\(\)](#). It validates and stores the user-supplied language key.

bool function InstallWizard::save_user() [line 1160]

validate and store the data for the first user account

This is the companion routine for [show_dialog_user\(\)](#). It stores the information about the first user account. We always store the data in the global `_SESSION`, even if something goes wrong. This makes that the user will use the latest values when re-doing the dialog.

We try to validate at least the password:

- minimum of 8 characters
- minimum 1 upper case, 1 lower case, 1 digit

Also, both username and full name should not be empty

void function InstallWizard::show_dialog_cancelled([\$m = ""]) [line 2126]

Function Parameters:

- *string \$m* margin for better readability of generated HTML-code

show the user that the process has been cancelled

this shows a screen with the message that the installation procedure has been cancelled. There is a single [OK] button that effectively allows the user to try again. The existing session (and all the data it might contain) is destroyed.

Note that we first construct the page (possibly in another language) in a separate variable before we destroy all data collected so far.

void function InstallWizard::show_dialog_cms([\$m = ""]) [line 811]

Function Parameters:

- *string \$m* margin for better readability of generated HTML-code

construct the dialog for essential cms data (title, paths, e-mail address)

This dialog contains the following fields:

- website_title (varchar(255))
- website_from_address (varchar(255))
- website_replyto_address (varchar(255))
- cms_dir (varchar(240))
- cms_www (varchar(240))
- cms_prokdir (varchar(240))
- cms_progwww (varchar(240))
- cms_dataroot (varchar(240))
- cms_demodata (boolean)
- cms_demodata_password (varchar(255) (but only a sha1() or md5() hash is stored eventually)

Some fields are suppressed in the dialog if the user selected a Standard installation:

- website_replyto_address: copied from website_from_address
- cms_prokdir: constructed from cms_dir
- cms_progwww: constructed from cms_www

Note the special trick with the test for a friendly URL; see [invisible_test_image\(\)](#) for the gory details.

- **TODO** can we suppress even more fields here in case of a Standard installation?
- **Uses** [InstallWizard::invisible_test_image\(\)](#)

void function InstallWizard::show_dialog_compatibility([\$m = ""]) [line 1265]

Function Parameters:

- *string \$m* margin for better readability of generated HTML-code

construct the comptibility overview

this routine displays a tabular overview of minimal compatibility requirements and the

current status/testresults. The table is constructed in a subroutine; we only deal with the display here.

Q: Why here, at the last stop before installation? A: Because we otherwise would leak information about our environment to complete strangers (assuming anyone can execute this script).

- **TODO** more tests to perform here: safe mode, memory limit, processing time limit, register globals

void function InstallWizard::show_dialog_confirm([\$m = ""]) [line 1487]

Function Parameters:

- *string* **\$m** margin for better readability of generated HTML-code

construct the overview/confirmation dialog

This dialog contains an overview of the information entered (excluding the passwords which are visually replaced with asterisks). This is the last chance the user gets to change the data entered. Once the user presses the [Next] button, the actual installation takes off.

void function InstallWizard::show_dialog_database([\$m = ""]) [line 536]

Function Parameters:

- *string* **\$m** margin for better readability of generated HTML-code

construct the dialog for database (server, host, username, password, etc.)

This dialog contains the following fields:

- db_type (pick from enumerated list)
- db_server (varchar(240))
- db_username (varchar(240))
- db_password (varchar(240))
- db_name (varchar(240))
- db_prefix (varchar(240))

One field is suppressed in the dialog if the user selected a Standard installation:

- db_prefix

`void function InstallWizard::show_dialog_finish([$m = "]) [line 2040]`

Function Parameters:

- *string* **\$m** margin for better readability of generated HTML-code

construct the finish screen

this dialog is displayed after a succesful installation. The user is prompted to select the next destination, which can be either admin.php (the backoffice), index.php (the frontpage), manual.php (the documentation) or the project's home page.

There is also an option to download config.php. Once the user's choice is submitted, the session is reset and config.php can no longer be downloaded. This session reset is done in the 'Done' dialog (see [end_session_and_redirect\(\)](#)).

Note that the boolean parameter `INSTALL['config_php_written']` indicates whether the file config.php was already succesfully written in the designated location. If this is the case, we show a different message compared to the case where the user still needs to download+upload config.php.

`void function InstallWizard::show_dialog_installtype([$m = "]) [line 352]`

Function Parameters:

- *string* **\$m** margin for better readability of generated HTML-code

construct the installtype + high visibility selection dialog

This dialog contains a radio button where the user selects 'standard' or 'custom' and also a checkbox for high visibility. As always the dialog ends with buttons to move forward, backward or to cancel the installation process altogether.

`void function InstallWizard::show_dialog_language([$m = "]) [line 269]`

Function Parameters:

- *string* **\$m** margin for better readability of generated HTML-code

construct the language selection dialog

this dialog allows the user to pick a language. The choices are determined by looking for translation files in the file system, specifically for files /program/install/languages/LL/install.php

where LL is the language code, see [get_list_of_install_languages\(\)](#).

`void function InstallWizard::show_dialog_license([$m = ""]) [line 462]`

Function Parameters:

- *string* **\$m** margin for better readability of generated HTML-code

construct a full license agreement and an input where the user must enter 'I agree'

This constructs a (long) license agreement dialog. We more or less force the user to actually scroll through it by having the input box + buttons after the agreement text. Note that the phrase to enter into the box is also translated, it may be 'I agree' in English but something else in other languages (done via translation file). If the user already accepted the license, the 'I agree'-text is already displayed in the dialog, in the current language. (A user could type 'I agree' in English, change the language into Dutch, in which case the \$value in the textbox would be something like 'Ik ga accoord' or whatever the translation of 'I agree' is.) This is done via an extra level of translation with the 'dialog_license_i_agree' translation in the prompt.

Also, the instruction to enter the exact words are repeated near the bottom of the screen, just to make sure the user understands what to do.

Note that the phrase the user enters is compared to the requested phrase in a more or less case-INsensitive way (see [check_license\(\)](#)).

`void function InstallWizard::show_dialog_user([$m = ""]) [line 1119]`

Function Parameters:

- *string* **\$m** margin for better readability of generated HTML-code

construct the dialog for the first user account

This dialog contains the following fields:

- user_username (varchar(255))
- user_full_name (varchar(255))
- user_email (varchar(255))
- user_password (varchar(255) (but only a sha1() or md5() hash is stored eventually)

An additional feature of this routine is to set a default email address for the account, by copying the address that can be found in the Reply-To: field that was entered earlier (or constructed from the From: address).

string function InstallWizard::t(\$key, [\$replace = "], [\$lang = "]) [*line 2774*]

Function Parameters:

- *string* **\$key** the key in the string-array with translations
- *array* **\$replace** key-value pairs used for search/replace in the translated string
- *string* **\$lang** indicates which language we should translate into

retrieve a translated string with optional parameters filled in

this routine tries to find a translated string based on the \$key. If \$replace is not empty, any keys in that array that are found in the translation are replace by the corresponding value.

The translations are read from the files /program/install/languages/LL/install.php, where LL is a language code (usually two-letter e.g. en, nl, de, fr). If the desired language is not available, English is used instead. If the requested translation is not found, the key of the translation is returned, sandwiched between the language codes. Usually this does not happen (all keys used have a translation), but it can be helpful when creating and testing a new translation.

Note that a language file is retrieved completely. This means that all the translations are read from file in one swoop; there is no need to go to the disk for every individual translation. The translation file is buffered in memory via the static variable \$phrases, which is an array keyed by language. That means that multiple languages can co-exist in this static variable. That last feature is used in constructing a list of available languages where the name of the language is expressed in the language itself (see [get_list_of_install_languages\(\)](#)).

Because the English language is the basis for all other languages, we always read the English translation the first time around, as a backup plan for missing prompts in other languages.

Note: By convention the keys in \$replace are upper case words, with optional underscores, sandwiched between curly braces. Examples: '{FIELD}', '{DATABASE}', '{RELEASE_DATE}'. The idea is that these words make life easier for translators.

TRUE function InstallWizard::validate(\$item, \$value) [*line 3130*]

Function Parameters:

- *array* **\$item** this array holds a field definition from a dialog definition
- *string* **\$value** this is the magic_unquote()'d and trim()'ed UTF-8 compliant string value the user POSTed

minimal validation of data input

this routine is a KISS-validator; we check for min/max stringlengths in strings and passwords (defaults: 0 and 255) and a valid item in listboxes. In case of error, a message is added to `$this->messages` and the function returns FALSE. On success we return TRUE. Note that additional validation could or should be done on some fields, e.g. a password of sufficient complexity, etc. This routine does not do that.

TRUE function InstallWizard::validate_password(\$label, \$password, [\$min_lower = 1], [\$min_upper = 1], [\$min_digit = 1]) [line 3206]

Function Parameters:

- *string* **\$label** this string holds the human-readable field name
- *string* **\$password** this is the `magic_unquote()`'d and `trim()`'ed password the user POSTed
- *int* **\$min_lower** the minimum number of lower case letters
- *int* **\$min_upper** the minimum number of upper case letters
- *int* **\$min_digit** the minimum number of digits

validation of password input

this routine analyses the password provided against the minimal requirements of password complexity. Here 'complexity' means a minimum number of upper case and lower case characters and a minimum number of digits. The required 'complexity' of a password is quite ASCII-centric. The idea is to make a user pick from a pool of at least 62 different characters rather than say 26 lowercase latin letters a-z. However, with UTF-8 the story is actually different: what is more "complex": "Aa1" or U+1F150

NEGATIVE CIRCLED LATIN CAPITAL LETTER A encoded as "\xF0\x9F\x85\x90"? Even though it is only 1 character, the result is byte string of length 4 rather than the meager 3-byte string that satisfies the minimum requirements...

Anyway. In order to take into account some additional upper case and lower case characters without resorting to loading the complete Unicode database, I opt for the following tricks. If `lowercase($str)` differs from `$str` itself, it was apparently not a lower case character but probably an upper case (or title case) character; good enough to count it toward the upper case character count. The same idea is used for the opposite: if `upper($str) != $str => $str` probably lowercase.

This algorithm works for A-Z and a-z but also for the 1100+ other Unicode characters that are case-aware. So, the user isn't held back if she insists on the password p a \xE1 \xBA \x9E w 0 r d (where \xE1 \xBA \x9E equates to U+1E9E LATIN CAPITAL LETTER SHARP S

Note that by default we still require at least 1 digit 0-9 and do not count any of the 410 "digits" like these zeros: U+0660: ARABIC-INDIC DIGIT ZERO U+06F0: EXTENDED ARABIC-INDIC DIGIT ZERO U+07C0: NKO DIGIT ZERO U+0966: DEVANAGARI DIGIT ZERO ... U+1D7EC: MATHEMATICAL SANS-SERIF BOLD DIGIT

ZERO U+1D7F6: MATHEMATICAL MONOSPACE DIGIT ZERO

Perhaps some other time... (Did I mention this is quite ASCII-centric?)

bool function InstallWizard::write_config_php() [*line 3308*]

attempt to write the file config.php in the correct location

this routine tries to write the file config.php. If this fails, we return FALSE, otherwise TRUE. Note that the permissions of the file are set to read-only (chmod 0400) for the owner of the file, and nothing for group and world. That should be enough for the webserver.

- **TODO** should we make the filemode (hardcoded at 0400) configurable/customisable?

Package waslang_ar Procedural Elements

demodata.php

/program/install/languages/ar/demodata.php

Language: ar (Ø§Ù,,Ø¹Ø±Ø¨ÙŠØ©) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_ar
- **Author** Said Taki < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.6 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/ar/install.php

Language: ar (Ø§Ù,,Ø¹Ø±Ø¨ÙŠØ©) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_ar
- **Author** Said Taki < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.6 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/ar/admin.php

Language: ar (Ø\$Ù,,Ø¹Ø±Ø¨Ù\$Ø©) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_ar
- **Author** Said Taki < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.6 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

ar_manifest.php

/program/languages/ar/ar_manifest.php - description of the Arabic translation

This file defines the Arabic language package ('ar'). This file is used when this package is installed.

- **Package** waslang_ar
- **Author** Said Taki < translators@websiteatschool.eu>
- **Version** \$Id: ar_manifest.php,v 1.7 2016/06/28 12:58:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/ar/loginlib.php

Language: ar (Ø\$Ù,,Ø¹Ø±Ø"ŮŠØ©) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_ar
- **Author** Said Taki < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.6 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/ar/was.php

Language: ar (Ø\$Ù,,Ø¹Ø±Ø¨ÙŠØ©) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_ar
- **Author** Said Taki < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.6 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/ar/htmlpage.php

Language: ar (Ø\$Ù,,Ø¹Ø±Ø¨Ù\$Ø©) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_ar
- **Author** Said Taki < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.6 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/ar/sitemap.php

Language: ar (Ø\$Ù,,Ø¹Ø±Ø"ŮŠØ©) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_ar
- **Author** Said Taki < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.6 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/ar/axis.php

Language: ar (Ø\$Ù,,Ø¹Ø±Ø¨ÙŠØ©) Release: 0.90.3 / 2012041700 (2012-04-17)

- **Package** waslang_ar
- **Author** Said Taki < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.4 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/ar/frugal.php

Language: ar (Ø\$Û,,Ø¹Ø±Ø¨ÛŠØ©) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_ar
- **Author** Said Taki < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.6 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/ar/rosalina.php

Language: ar (Ø§Ù,,Ø¹Ø±Ø¨ÙŠØ©) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_ar
- **Author** Said Taki < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.6 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/ar/schoolyard.php

Language: ar (Ø§Ù,,Ø¹Ø±Ø¨ÙØ©) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_ar
- **Author** Said Taki < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.6 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_bg Procedural Elements

demodata.php

/program/install/languages/bg/demodata.php

Language: bg (Ð±ÑŠÐ»Ð³Ð°Ñ•Ð°Ð,) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_bg
- **Author** Boyan Kirchev < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.2 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/bg/install.php

Language: bg (Български език) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_bg
- **Author** Boyan Kirchev < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.2 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/bg/admin.php

Language: bg (Български език) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_bg
- **Author** Galina Valina < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.3 2016/06/28 12:58:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

bg_manifest.php

/program/languages/bg/bg_manifest.php - description of the Bulgarian translation

This file defines the Bulgarian language package ('bg'). This file is used when this package is installed.

- **Package** waslang_bg
- **Author** Boyan Kirchev, Galina Valina < translators@websiteatschool.eu >
- **Version** \$Id: bg_manifest.php,v 1.3 2016/06/28 12:58:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/bg/loginlib.php

Language: bg (Български език) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_bg
- **Author** Galina Valina < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.3 2016/06/28 12:58:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/bg/was.php

Language: bg (Ð±ÑŠÐ»Ð³Ð°Ñ€Ð°Ð¸) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_bg
- **Author** Galina Valina < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.3 2016/06/28 12:58:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/bg/aggregator.php

Language: bg (Български език) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_bg
- **Author** Galina Valina < translators@websiteatschool.eu>
- **Version** \$Id: aggregator.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

althing.php

/program/modules/althing/languages/bg/althing.php

Language: bg (Български език) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_bg
- **Author** Galina Valina < translators@websiteatschool.eu>
- **Version** \$Id: althing.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/bg/crew.php

Language: bg (Български език) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_bg
- **Author** Boyan Kirchev < translators@websiteatschool.eu>
- **Version** \$Id: crew.php,v 1.2 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/bg/htmlpage.php

Language: bg (Български език) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_bg
- **Author** Boyan Kirchev < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.2 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage.php

/program/modules/mailpage/languages/bg/mailpage.php

Language: bg (Български език) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_bg
- **Author** Boyan Kirchev < translators@websiteatschool.eu>
- **Version** \$Id: mailpage.php,v 1.2 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/bg/redirect.php

Language: bg (Български) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_bg
- **Author** Boyan Kirchev < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.2 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/bg/sitemap.php

Language: bg (Български език) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_bg
- **Author** Galina Valina < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots.php

/program/modules/snapshots/languages/bg/snapshots.php

Language: bg (Български език) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_bg
- **Author** Boyan Kirchev < translators@websiteatschool.eu>
- **Version** \$Id: snapshots.php,v 1.2 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/bg/axis.php

Language: bg (Ð±ÑŠÐ»Ð³Ð°Ñ•Ð°) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_bg
- **Author** Galina Valina < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/bg/cornelia.php

Language: bg (Български език) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_bg
- **Author** Boyan Kirchev < translators@websiteatschool.eu>
- **Version** \$Id: cornelia.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/bg/frugal.php

Language: bg (Български език) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_bg
- **Author** Boyan Kirchev < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/bg/rosalina.php

Language: bg (Български език) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_bg
- **Author** Boyan Kirchev < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.2 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/bg/schoolyard.php

Language: bg (Български език) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_bg
- **Author** Boyan Kirchev < translators@websiteatschool.eu >
- **Version** \$Id: schoolyard.php,v 1.2 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php

/program/themes/sophia/languages/bg/sophia.php

Language: bg (Български език) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_bg
- **Author** Boyan Kirchev < translators@websiteatschool.eu >
- **Version** \$Id: sophia.php,v 1.2 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_da Procedural Elements

demodata.php

/program/install/languages/da/demodata.php

Language: da (Dansk) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_da
- **Author** Christian Borum Loebner - Olesen < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.6 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/da/install.php

Language: da (Dansk) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_da
- **Author** Christian Borum Loebner - Olesen < translators@websiteatschool.eu >
- **Version** \$Id: install.php,v 1.8 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/da/admin.php

Language: da (Dansk) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_da
- **Author** Christian Borum Loebner - Olesen < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.6 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

da_manifest.php

/program/languages/da/da_manifest.php - description of the Danish translation

This file defines the Danish language package ('da'). This file is used when this package is installed.

- **Package** waslang_da
- **Author** Christian Borum Loebner - Olesen < translators@websiteatschool.eu>
- **Version** \$Id: da_manifest.php,v 1.7 2016/06/28 12:58:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/da/loginlib.php

Language: da (Dansk) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_da
- **Author** Christian Borum Loebner - Olesen < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.6 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/da/was.php

Language: da (Dansk) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_da
- **Author** Christian Borum < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.6 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/da/htmlpage.php

Language: da (Dansk) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_da
- **Author** Christian Borum Loebner - Olesen < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.6 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/da/sitemap.php

Language: da (Dansk) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_da
- **Author** Christian Borum Loebner - Olesen < translators@websiteatschool.eu >
- **Version** \$Id: sitemap.php,v 1.6 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/da/axis.php

Language: da (Dansk) Release: 0.90.3 / 2012041700 (2012-04-17)

- **Package** waslang_da
- **Author** Christian Borum Loebner - Olesen < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.4 2016/03/23 09:28:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php
/program/themes/frugal/languages/da/frugal.php
Language: da (Dansk) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_da
- **Author** Christian Borum Loebner - Olesen < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.6 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/da/rosalina.php

Language: da (Dansk) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_da
- **Author** Christian Borum < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.6 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/da/schoolyard.php

Language: da (Dansk) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_da
- **Author** Christian Borum Loebner - Olesen < translators@websiteatschool.eu >
- **Version** \$Id: schoolyard.php,v 1.6 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_de Procedural Elements

demodata.php

/program/install/languages/de/demodata.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** Fabienne Kudzielka < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.6 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/de/install.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** Fabienne Kudzielka < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.6 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/de/admin.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** C.GÃ¶hnert < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.7 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

de_manifest.php

/program/languages/de/de_manifest.php - description of the German translation

This file defines the German language package ('de'). This file is used when this package is installed.

- **Package** waslang_de
- **Author** DavidP, S. Stadoll, Fabienne Kudzielka, Claudia GÃ¶hnert
< translators@websiteatschool.eu >
- **Version** \$Id: de_manifest.php,v 1.8 2016/06/28 12:58:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/de/loginlib.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** David Prousch < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.7 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/de/was.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** David Prousch < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.7 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/de/aggregator.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** Claudia GÃ¶hnert < translators@websiteatschool.eu >
- **Version** \$Id: aggregator.php,v 1.2 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/de/crew.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** Claudia GÃ¶hnert < translators@websiteatschool.eu >
- **Version** \$Id: crew.php,v 1.2 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/de/htmlpage.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** David < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.7 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage.php

/program/modules/mailpage/languages/de/mailpage.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** Claudia GÃ¶hnert < translators@websiteatschool.eu >
- **Version** \$Id: mailpage.php,v 1.2 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/de/redirect.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** Claudia GÃ¶hnert < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.5 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/de/sitemap.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** Claudia GÃ¶hnert < translators@websiteatschool.eu >
- **Version** \$Id: sitemap.php,v 1.6 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots.php

/program/modules/snapshots/languages/de/snapshots.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** Claudia GÃ¶hnert < translators@websiteatschool.eu >
- **Version** \$Id: snapshots.php,v 1.2 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/de/axis.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** Claudia GÃ¶hnert < translators@websiteatschool.eu >
- **Version** \$Id: axis.php,v 1.6 2016/03/23 09:28:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/de/cornelia.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** Claudia GÃ¶hnert < translators@websiteatschool.eu>
- **Version** \$Id: cornelia.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/de/frugal.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** C.GÃ¶hnert < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.7 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/de/rosalina.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** Claudia GÃ¶hnert < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.4 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/de/schoolyard.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** Claudia GÃ¶hnert < translators@websiteatschool.eu >
- **Version** \$Id: schoolyard.php,v 1.6 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php

/program/themes/sophia/languages/de/sophia.php

Language: de (Deutsch) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_de
- **Author** Claudia GÃ¶hnert < translators@websiteatschool.eu>
- **Version** \$Id: sophia.php,v 1.2 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_el Procedural Elements

demodata.php

/program/install/languages/el/demodata.php

Language: el (Γ•Γ»Γ»Γ·Γ½Γ¹ΓºΓ¬) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_el
- **Author** Iakovos Christoforidis < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.4 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/el/install.php

Language: el (ἑλῆνικὰ) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_el
- **Author** Iakovos Christoforidis < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.5 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/el/admin.php

Language: el (•»»½»-) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_el
- **Author** Niki Papachristou < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.6 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

el_manifest.php

/program/languages/el/el_manifest.php - description of the Greek translation

This file defines the Greek language package ('el'). This file is used when this package is installed.

- **Package** waslang_el
- **Author** Iakovos Christoforidis, Niki Papachristou < translators@websiteatschool.eu>
- **Version** \$Id: el_manifest.php,v 1.5 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/el/loginlib.php

Language: el (Γ•Γ»Γ»Γ·Γ½Γ¹ΓºΓ¬) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_el
- **Author** Niki Papachristou < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.5 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/el/was.php

Language: el (␣•␣»␣»␣␣½␣␣␣␣→) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_el
- **Author** Niki Papachristou < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.5 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/el/aggregator.php

Language: el (␣•␣»␣»␣␣½␣␣␣␣␣␣␣) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_el
- **Author** Niki Papachristou < translators@websiteatschool.eu>
- **Version** \$Id: aggregator.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

althing.php

/program/modules/althing/languages/el/althing.php

Language: el (Γ•Γ»Γ»Γ·Γ½Γ¹ΓºΓ¬) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_el
- **Author** Niki Papachristou < translators@websiteatschool.eu>
- **Version** \$Id: althing.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

confab.php

/program/modules/confab/languages/el/confab.php

Language: el (␣•␣»␣»␣␣½␣␣␣␣␣␣␣) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_el
- **Author** Niki Papachristou < translators@websiteatschool.eu>
- **Version** \$Id: confab.php,v 1.1 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/el/crew.php

Language: el (î•î»î»î·î½î¹îºî¬) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_el
- **Author** Niki Papachristou < translators@websiteatschool.eu>
- **Version** \$Id: crew.php,v 1.1 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/el/htmlpage.php

Language: el (␣•␣»␣»␣␣½␣␣␣␣␣␣␣) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_el
- **Author** Niki Papachristou < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.5 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/el/redirect.php

Language: el (␣•␣»␣»␣␣½␣␣␣␣␣␣␣) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_el
- **Author** Niki Papachristou < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.1 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/el/sitemap.php

Language: el (␣•␣»␣»␣␣½␣␣␣␣→) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_el
- **Author** Niki Papachristou < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.5 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/el/axis.php

Language: el (Î•Î»Î»Î»Î½Î°Î¬) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_el
- **Author** Iakovos Christoforidis < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.4 2016/03/23 09:28:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/el/frugal.php

Language: el (ἰ•ἰ»ἰ»ἰ·ἰ½ἰ¹ἰ°ἰ¬) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_el
- **Author** Iakovos Christoforidis < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.4 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/el/rosalina.php

Language: el (Î•Î»Î»Î»Î½Î°Î¬) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_el
- **Author** Iakovos Christoforidis < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.4 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/el/schoolyard.php

Language: el (ἰ•ἰ»ἰ»ἰ·ἰ½ἰ¹ἰ°ἰ¬) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_el
- **Author** Iakovos Christoforidis < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.4 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_en Procedural Elements

en_manifest.php

/program/languages/en/en_manifest.php - description of the main language/translation (English)

This file defines the English language package ('en'). This file is used when this essential (core) package is installed.

- **Package** waslang_en
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: en_manifest.php,v 1.13 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_es Procedural Elements

demodata.php

/program/install/languages/es/demodata.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Anouk Coumans & Hanna Tulleken < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.10 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/es/install.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Hanna Tulleken < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.12 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/es/admin.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Teresa Villanueva < translators@websiteatschool.eu >
- **Version** \$Id: admin.php,v 1.12 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

es_manifest.php

/program/languages/es/es_manifest.php - description of the Spanish translation

This file defines the Spanish language package ('es'). This file is used when this package is installed.

- **Package** waslang_es
- **Author** Anouk Coumans, Hanna Tulleken, Margot Molier < translators@websiteatschool.eu >
- **Version** \$Id: es_manifest.php,v 1.11 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/es/loginlib.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Teresa Villamán < translators@websiteatschool.eu >
- **Version** \$Id: loginlib.php,v 1.10 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/es/was.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Teresa Villamán < translators@websiteatschool.eu >
- **Version** \$Id: was.php,v 1.10 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/es/aggregator.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Hannah Tulleken < translators@websiteatschool.eu>
- **Version** \$Id: aggregator.php,v 1.2 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/es/crew.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Hannah Tulleken < translators@websiteatschool.eu>
- **Version** \$Id: crew.php,v 1.2 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/es/htmlpage.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Anouk Coumans < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.10 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/es/sitemap.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Anouk Coumans < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.8 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/es/axis.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Anouk Coumans < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.6 2016/03/23 09:28:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/es/cornelia.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Hannah Tulleken < translators@websiteatschool.eu>
- **Version** \$Id: cornelia.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/es/frugal.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Margot Molier < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.10 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/es/rosalina.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Anouk Coumans < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.9 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/es/schoolyard.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Anouk Coumans < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.8 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php

/program/themes/sophia/languages/es/sophia.php

Language: es (Español) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_es
- **Author** Hannah Tulleken < translators@websiteatschool.eu>
- **Version** \$Id: sophia.php,v 1.2 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_fa Procedural Elements

demodata.php

/program/install/languages/fa/demodata.php

Language: fa (â€«Ù•Ø§Ø±Ø³Ùœ) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_fa
- **Author** A. Darvishi < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.8 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/fa/install.php

Language: fa (â€«Ù•Ø§Ø±Ø³Ùœ) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_fa
- **Author** A.Darvishi < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.8 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/fa/admin.php

Language: fa (â€«Ù•Ø§Ø±Ø³Ùœ) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_fa
- **Author** Pegah Mohtadi Tabrizi < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.10 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

fa_manifest.php

/program/languages/fa/fa_manifest.php - description of the Persian translation

This file defines the Persian language package ('fa'). This file is used when this package is installed.

- **Package** waslang_fa
- **Author** A. Darvishi, P.M. Tabrizi < translators@websiteatschool.eu >
- **Version** \$Id: fa_manifest.php,v 1.7 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/fa/loginlib.php

Language: fa (â€«Ù•Ø§Ø±Ø³Ùœ) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_fa
- **Author** Pegah Mohtadi Tabrizi < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.8 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/fa/was.php

Language: fa (â€«Û•Ø§Ø±Ø³Ûœ) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_fa
- **Author** Pegah Mohtadi Tabrizi < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.7 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/fa/htmlpage.php

Language: fa (â€«Ù•Ø§Ø±Ø³Ùœ) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_fa
- **Author** A. Darvishi < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.8 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/fa/sitemap.php

Language: fa (â€«Ù•Ø§Ø±Ø³Ùœ) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_fa
- **Author** A. Darvishi < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.8 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/fa/axis.php

Language: fa (â€«Ù•Ø§Ø±Ø³Ùœ) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_fa
- **Author** A. Darvishi < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.8 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/fa/frugal.php

Language: fa (â€«Ù•Ø§Ø±Ø³Ùœ) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_fa
- **Author** A. Darvishi < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.8 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/fa/rosalina.php

Language: fa (â€«Ù•Ø§Ø±Ø³Ùœ) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_fa
- **Author** A. Darvishi < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.8 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/fa/schoolyard.php

Language: fa (â€«Ù•Ø§Ø±Ø³Ùœ) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_fa
- **Author** A. Darvishi < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.8 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_fi Procedural Elements

demodata.php

/program/install/languages/fi/demodata.php

Language: fi (Suomi) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_fi
- **Author** Laura R  man < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.4 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/fi/install.php

Language: fi (Suomi) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_fi
- **Author** Laura R  man < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.4 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/fi/admin.php

Language: fi (Suomi) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_fi
- **Author** Laura R  man < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.5 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

fi_manifest.php

/program/languages/fi/fi_manifest.php - description of the Finnish translation

This file defines the Finnish language package ('fi'). This file is used when this package is installed.

- **Package** waslang_fi
- **Author** Laura R  man < translators@websiteatschool.eu >
- **Version** \$Id: fi_manifest.php,v 1.5 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/fi/loginlib.php

Language: fi (Suomi) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_fi
- **Author** Laura R  man < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.4 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/fi/was.php

Language: fi (Suomi) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_fi
- **Author** Laura R  man < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.4 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/fi/htmlpage.php

Language: fi (Suomi) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_fi
- **Author** Laura R  man < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.4 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/fi/sitemap.php

Language: fi (Suomi) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_fi
- **Author** Laura R  man < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.4 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/fi/axis.php

Language: fi (Suomi) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_fi
- **Author** Laura R  man < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.4 2016/03/23 09:28:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/fi/frugal.php

Language: fi (Suomi) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_fi
- **Author** Laura R  man < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.4 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/fi/rosalina.php

Language: fi (Suomi) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_fi
- **Author** Laura R  man < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.4 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/fi/schoolyard.php

Language: fi (Suomi) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_fi
- **Author** Laura R  man < translators@websiteatschool.eu >
- **Version** \$Id: schoolyard.php,v 1.4 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_fil Procedural Elements

admin.php

/program/languages/fil/admin.php

Language: fil (Filipino) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fil
- **Author** Abigail Hieselaar < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

fil_manifest.php

/program/languages/fil/fil_manifest.php - description of the Filipino translation

This file defines the Filipino language package ('fil'). This file is used when this package is installed.

- **Package** waslang_fil
- **Author** Abigail Hieselaar < translators@websiteatschool.eu >
- **Version** \$Id: fil_manifest.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/fil/loginlib.php

Language: fil (Filipino) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fil
- **Author** Abigail Hieselaar < translators@websiteatschool.eu >
- **Version** \$Id: loginlib.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/fil/was.php

Language: fil (Filipino) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fil
- **Author** Abigail Hieselaar < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_fr Procedural Elements

demodata.php

/program/install/languages/fr/demodata.php

Language: fr (FranÃ§ais) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: demodata.php,v 1.9 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/fr/install.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: install.php,v 1.9 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/fr/admin.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: admin.php,v 1.10 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

fr_manifest.php

/program/languages/fr/fr_manifest.php - description of the French translation

This file defines the French language package ('fr'). This file is used when this package is installed.

- **Package** waslang_fr
- **Author** Jean Peyratout < jean.peyratout@abul.org> Marjolaine Audoux
< translators@websiteatschool.eu>
- **Version** \$Id: fr_manifest.php,v 1.9 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/fr/loginlib.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: loginlib.php,v 1.8 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/fr/was.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: was.php,v 1.8 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/fr/aggregator.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: aggregator.php,v 1.2 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/fr/crew.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: crew.php,v 1.2 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/fr/htmlpage.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: htmlpage.php,v 1.8 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage.php

/program/modules/mailpage/languages/fr/mailpage.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: mailpage.php,v 1.2 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/fr/redirect.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: redirect.php,v 1.5 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/fr/sitemap.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: sitemap.php,v 1.7 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots.php

/program/modules/snapshots/languages/fr/snapshots.php

Language: fr (Fran ais) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: snapshots.php,v 1.2 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/fr/axis.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Marjolaine Audoux < translators@websiteatschool.eu>
< jean.peyratout@abul.org>
- **Version** \$Id: axis.php,v 1.6 2016/03/23 09:28:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/fr/cornelia.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: cornelia.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/fr/frugal.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: frugal.php,v 1.8 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/fr/rosalina.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: rosalina.php,v 1.8 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/fr/schoolyard.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: schoolyard.php,v 1.7 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php

/program/themes/sophia/languages/fr/sophia.php

Language: fr (Français) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fr
- **Author** Jean Peyratout < translators@websiteatschool.eu > < jean.peyratout@abul.org >
- **Version** \$Id: sophia.php,v 1.2 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_fry Procedural Elements

demodata.php

/program/install/languages/fry/demodata.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.3 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/fry/install.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.3 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/fry/admin.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

fry_manifest.php

/program/languages/fry/fry_manifest.php - description of the Western Frisian translation

This file defines the Western Frisian language package ('fry'). This file is used when this package is installed.

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: fry_manifest.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/fry/loginlib.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/fry/was.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/fry/aggregator.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: aggregator.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

althing.php

/program/modules/althing/languages/fry/althing.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: althing.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

confab.php

/program/modules/confab/languages/fry/confab.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: confab.php,v 1.1 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/fry/crew.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: crew.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/fry/htmlpage.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage.php

/program/modules/mailpage/languages/fry/mailpage.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: mailpage.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/fry/redirect.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/fry/sitemap.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots.php

/program/modules/snapshots/languages/fry/snapshots.php

Language: fry (Frysk) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: snapshots.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/fry/axis.php

Language: fry (Frysk) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.2 2016/03/23 09:28:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/fry/cornelia.php

Language: fry (Frysk) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: cornelia.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/fry/frugal.php

Language: fry (Frysk) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.2 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/fry/rosalina.php

Language: fry (Frysk) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.2 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/fry/schoolyard.php

Language: fry (Frysk) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.2 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php

/program/themes/sophia/languages/fry/sophia.php

Language: fry (Frysk) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_fry
- **Author** Piet Damsma < translators@websiteatschool.eu>
- **Version** \$Id: sophia.php,v 1.2 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

demodata.php

Language: hi (à¤¹à¤¸à¤¸à¤¸à¤¸) Release: 0.90.6 / 2016062900 (2016-06-29)

-
- Generated by phpDocumentor v1.4.0 <http://www.phpdoc.org> - <http://pear.php.net/package/PhpDocumentor> - <http://www.sourceforge.net/projects/phpdocu> Page 677 of 1414

/program/languages/hi/admin.php

Language: hi (àà¹àà¸àà,àà¹àà¥€) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_hi
- **Author** Rashmi Bookanakere < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

hi_manifest.php

/program/languages/hi/hi_manifest.php - description of the Hindi translation

This file defines the Hindi language package ('hi'). This file is used when this package is installed.

- **Package** waslang_hi
- **Author** Rashmi Bookanakere < translators@websiteatschool.eu>
- **Version** \$Id: hi_manifest.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

[/program/languages/hi/loginlib.php](#)

Language: hi (àà¹àà¿àà,àà¹àà¥€) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_hi
- **Author** Rashmi Bookanakere < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

</program/languages/hi/was.php>

Language: hi (à¤¢¹à¤¸ à¤²à¤¿à¥€) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_hi
- **Author** Rashmi Bookanakere < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php
/program/modules/htmlpage/languages/hi/htmlpage.php

/program/modules/htmlpage/languages/hi/htmlpage.php
Language: hi (à¹à; à², à³à¥€) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_hi
- **Author** Rashmi Bookanakere < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.1 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php
/program/themes/sophia/languages/hi/sophia.php

/program/themes/sophia/languages/hi/sophia.php

- Language: hi (à¤¤à¤¤à¤¤,à¤¤à¤¤à¤¤) Release: 0.90.6 / 2016062900 (2016-06-29)

Package waslang_hu Procedural Elements

demodata.php

/program/install/languages/hu/demodata.php

Language: hu (Magyar) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_hu
- **Author** Erika Swiderski < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.7 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/hu/install.php

Language: hu (Magyar) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_hu
- **Author** Erika Swiderski, Gergely Sipos < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.9 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/hu/admin.php

Language: hu (Magyar) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_hu
- **Author** Gergely Sipos, Erika Swiderski < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.8 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

hu_manifest.php

/program/languages/hu/hu_manifest.php - description of the Hungarian translation

This file defines the Hungarian language package ('hu'). This file is used when this package is installed.

- **Package** waslang_hu
- **Author** Erika Swiderski, Gergely Sipos < translators@websiteatschool.eu>
- **Version** \$Id: hu_manifest.php,v 1.7 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/hu/loginlib.php

Language: hu (Magyar) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_hu
- **Author** Erika Swiderski < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.7 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/hu/was.php

Language: hu (Magyar) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_hu
- **Author** Erika Swiderski < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.7 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/hu/htmlpage.php

Language: hu (Magyar) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_hu
- **Author** Erika Swiderski < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.7 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/hu/sitemap.php

Language: hu (Magyar) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_hu
- **Author** Erika Swiderski < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.7 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/hu/axis.php

Language: hu (Magyar) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_hu
- **Author** Gergely Sipos < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.7 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/hu/frugal.php

Language: hu (Magyar) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_hu
- **Author** Erika Swiderski < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.7 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/hu/rosalina.php

Language: hu (Magyar) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_hu
- **Author** Gergely Sipos, Erika Swiderski < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.7 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/hu/schoolyard.php

Language: hu (Magyar) Release: 0.90.5 / 2016062900 (2016-06-29)

- **Package** waslang_hu
- **Author** Erika Swiderski, Gergely Sipos < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.7 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_it Procedural Elements

demodata.php

/program/install/languages/it/demodata.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: demodata.php,v 1.2 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/it/install.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemüller < translators@websiteatschool.eu >
- **Version** \$Id: install.php,v 1.2 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/it/admin.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: admin.php,v 1.2 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

it_manifest.php

/program/languages/it/it_manifest.php - description of the Italian translation

This file defines the Italian language package ('it'). This file is used when this package is installed.

- **Package** waslang_it
- **Author** Ernst Frankemölle, Rosanna Gaddoni < translators@websiteatschool.eu >
- **Version** \$Id: it_manifest.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/it/loginlib.php

Language: it (Italiano) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_it
- **Author** Rosanna Gaddoni < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/it/was.php

Language: it (Italiano) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_it
- **Author** Rosanna Gaddoni < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/it/aggregator.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: aggregator.php,v 1.2 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/it/crew.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: crew.php,v 1.2 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/it/htmlpage.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: htmlpage.php,v 1.2 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage.php

/program/modules/mailpage/languages/it/mailpage.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: mailpage.php,v 1.2 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/it/redirect.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: redirect.php,v 1.2 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/it/sitemap.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: sitemap.php,v 1.2 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots.php

/program/modules/snapshots/languages/it/snapshots.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: snapshots.php,v 1.2 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/it/axis.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: axis.php,v 1.2 2016/03/23 09:28:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/it/cornelia.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: cornelia.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/it/frugal.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: frugal.php,v 1.2 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/it/rosalina.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: rosalina.php,v 1.2 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/it/schoolyard.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: schoolyard.php,v 1.2 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php

/program/themes/sophia/languages/it/sophia.php

Language: it (Italiano) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_it
- **Author** Ernst Frankemölle < translators@websiteatschool.eu >
- **Version** \$Id: sophia.php,v 1.2 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_nl Procedural Elements

nl_manifest.php

/program/languages/nl/nl_manifest.php - description of the Dutch translation

This file defines the Dutch language package ('nl'). This file is used when this package is installed.

- **Package** waslang_nl
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: nl_manifest.php,v 1.13 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_pap Procedural Elements

demodata.php

/program/install/languages/pap/demodata.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.2 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/pap/install.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.2 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/pap/admin.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.2 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/pap/loginlib.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.2 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

pap_manifest.php

/program/languages/pap/pap_manifest.php - description of the Papiamento translation

This file defines the Papiamento language package ('pap'). This file is used when this package is installed.

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: pap_manifest.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/pap/was.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.2 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/pap/aggregator.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: aggregator.php,v 1.2 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/pap/crew.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: crew.php,v 1.2 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/pap/htmlpage.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.2 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage.php

/program/modules/mailpage/languages/pap/mailpage.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: mailpage.php,v 1.2 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/pap/redirect.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.2 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/pap/sitemap.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.2 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots.php

/program/modules/snapshots/languages/pap/snapshots.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: snapshots.php,v 1.2 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/pap/axis.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.2 2016/03/23 09:28:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/pap/cornelia.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: cornelia.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/pap/frugal.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.2 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/pap/rosalina.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.2 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/pap/schoolyard.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.2 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php

/program/themes/sophia/languages/pap/sophia.php

Language: pap (Papiamentu) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pap
- **Author** Giovanni Thomas < translators@websiteatschool.eu>
- **Version** \$Id: sophia.php,v 1.2 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_pl Procedural Elements

demodata.php

/program/install/languages/pl/demodata.php

Language: pl (Polski) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_pl
- **Author** Waldemar Pankiw < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.7 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/pl/install.php

Language: pl (Polski) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_pl
- **Author** Waldemar Pankiw < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.7 2016/03/23 09:28:33 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/pl/admin.php

Language: pl (Polski) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_pl
- **Author** Waldemar Pankiw < translators@websiteatschool.eu >
- **Version** \$Id: admin.php,v 1.8 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/pl/loginlib.php

Language: pl (Polski) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_pl
- **Author** Waldemar Pankiw < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.7 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

pl_manifest.php

/program/languages/pl/pl_manifest.php - description of the Polish translation

This file defines the Polish language package ('pl'). This file is used when this package is installed.

- **Package** waslang_pl
- **Author** Waldemar Pankiw < translators@websiteatschool.eu >
- **Version** \$Id: pl_manifest.php,v 1.8 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/pl/was.php

Language: pl (Polski) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_pl
- **Author** Waldemar Pankiw < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.7 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/pl/htmlpage.php

Language: pl (Polski) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_pl
- **Author** Waldemar Pankiw < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.7 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/pl/sitemap.php

Language: pl (Polski) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_pl
- **Author** Waldemar Pankiw < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.7 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/pl/axis.php

Language: pl (Polski) Release: 0.90.3 / 2012041700 (2012-04-17)

- **Package** waslang_pl
- **Author** Waldemar Pankiw < translators@websiteatschool.eu >
- **Version** \$Id: axis.php,v 1.4 2016/03/23 09:28:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/pl/frugal.php
Language: pl (Polski) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_pl
- **Author** Waldemar Pankiw < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.7 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/pl/rosalina.php

Language: pl (Polski) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_pl
- **Author** Waldemar Pankiw < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.8 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/pl/schoolyard.php

Language: pl (Polski) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_pl
- **Author** Waldemar Pankiw < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.7 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_pt Procedural Elements

demodata.php

/program/install/languages/pt/demodata.php

Language: pt (PortuguÃas) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Cristina Tracz < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.6 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/pt/install.php

Language: pt (PortuguÃas) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Cristina Tracz < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.6 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/pt/admin.php

Language: pt (PortuguÃas) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Cristina Tracz < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.7 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/pt/loginlib.php

Language: pt (Portugu s) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Cristina Tracz < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.6 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

pt_manifest.php

/program/languages/pt/pt_manifest.php - description of the Portugese translation

This file defines the Portugese language package ('pt'). This file is used when this package is installed.

- **Package** waslang_pt
- **Author** Rita Valente Ribeiro da Silva < translators@websiteatschool.eu >
- **Version** \$Id: pt_manifest.php,v 1.7 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/pt/was.php

Language: pt (PortuguÃas) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Thais Rizzi < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.6 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/pt/aggregator.php

Language: pt (PortuguÃas) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Tracz < translators@websiteatschool.eu>
- **Version** \$Id: aggregator.php,v 1.2 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/pt/crew.php

Language: pt (PortuguÃas) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Tracz < translators@websiteatschool.eu>
- **Version** \$Id: crew.php,v 1.2 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/pt/htmlpage.php

Language: pt (Portugu s) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Rita Valente Ribeiro da Silva < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.6 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage.php

/program/modules/mailpage/languages/pt/mailpage.php

Language: pt (Portugu s) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Tracz < translators@websiteatschool.eu>
- **Version** \$Id: mailpage.php,v 1.2 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/pt/redirect.php

Language: pt (PortuguÃas) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Tracz < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.5 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/pt/sitemap.php

Language: pt (Portugu s) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Cristina Tracz < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.6 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots.php

/program/modules/snapshots/languages/pt/snapshots.php

Language: pt (PortuguÃas) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Tracz < translators@websiteatschool.eu>
- **Version** \$Id: snapshots.php,v 1.2 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/pt/axis.php

Language: pt (Portugu s) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Tracz < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.5 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/pt/cornelia.php
Language: pt (PortuguÃas) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Tracz < translators@websiteatschool.eu>
- **Version** \$Id: cornelia.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/pt/frugal.php

Language: pt (PortuguÃas) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Cristina Tracz < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.6 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/pt/rosalina.php

Language: pt (PortuguÃas) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Cristina Tracz < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.6 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/pt/schoolyard.php

Language: pt (PortuguÃas) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Cristina Tracz < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.6 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php

/program/themes/sophia/languages/pt/sophia.php

Language: pt (PortuguÃas) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_pt
- **Author** Keli Tracz < translators@websiteatschool.eu>
- **Version** \$Id: sophia.php,v 1.2 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_ro Procedural Elements

admin.php

/program/languages/ro/admin.php

Language: ro (Românește™) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ro
- **Author** Loana Lenghel < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.2 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/ro/loginlib.php

Language: ro (Românește™) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ro
- **Author** Loana Lenghel < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.2 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

ro_manifest.php

/program/languages/ro/ro_manifest.php - description of the Romanian translation

This file defines the Romanian language package ('ro'). This file is used when this package is installed.

- **Package** waslang_ro
- **Author** Loana Lenghel < translators@websiteatschool.eu>
- **Version** \$Id: ro_manifest.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/ro/was.php

Language: ro (Românește™) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ro
- **Author** Loana Lenghel < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.2 2016/03/23 09:28:20 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/ro/htmlpage.php

Language: ro (Românește™) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ro
- **Author** Iulia Modi < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.2 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/ro/sitemap.php

Language: ro (Românește™) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ro
- **Author** Iulia Modi < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.2 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/ro/axis.php

Language: ro (Românește™) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ro
- **Author** Iulia Modi < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/ro/frugal.php

Language: ro (Românește™) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ro
- **Author** Iulia Modi < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.2 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/ro/rosalina.php

Language: ro (Românește™) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ro
- **Author** Iulia Modi < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.2 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_ru Procedural Elements

demodata.php

/program/install/languages/ru/demodata.php

Language: ru (Ð ÑfÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.4 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/ru/install.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.4 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/ru/admin.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.5 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/ru/loginlib.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.4 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

ru_manifest.php

/program/languages/ru/ru_manifest.php - description of the Russian translation

This file defines the Russian language package ('ru'). This file is used when this package is installed.

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu >
- **Version** \$Id: ru_manifest.php,v 1.5 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/ru/was.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.4 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/ru/aggregator.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: aggregator.php,v 1.2 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/ru/crew.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: crew.php,v 1.2 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/ru/htmlpage.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.4 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage.php

/program/modules/mailpage/languages/ru/mailpage.php

Language: ru (Ð ÑfÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: mailpage.php,v 1.2 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/ru/redirect.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.2 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/ru/sitemap.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.4 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots.php

/program/modules/snapshots/languages/ru/snapshots.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: snapshots.php,v 1.2 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/ru/axis.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.4 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/ru/cornelia.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: cornelia.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/ru/frugal.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.4 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/ru/rosalina.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.4 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/ru/schoolyard.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.4 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php

/program/themes/sophia/languages/ru/sophia.php

Language: ru (Ð ÑƒÑ•Ñ•Ð°Ð, Ð¹) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ru
- **Author** Anastassia Blechko < translators@websiteatschool.eu>
- **Version** \$Id: sophia.php,v 1.2 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_sh Procedural Elements

admin.php

/program/languages/sh/admin.php

Language: sh (Srpsko-Hrvatski) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_sh
- **Author** Sladjana Cetin < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/sh/loginlib.php

Language: sh (Srpsko-Hrvatski) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_sh
- **Author** Sladjana Cetin < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sh_manifest.php

/program/languages/sh/sh_manifest.php - description of the Serbo-Croatian translation

This file defines the Serbo-Croatian language package ('sh'). This file is used when this package is installed.

- **Package** waslang_sh
- **Author** Sladjana Cetin < translators@websiteatschool.eu>
- **Version** \$Id: sh_manifest.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/sh/was.php

Language: sh (Srpsko-Hrvatski) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_sh
- **Author** Sladjana Cetin < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_sk Procedural Elements

admin.php

/program/languages/sk/admin.php

Language: sk (Slovak) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_sk
- **Author** Nikoleta Koncikova < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/sk/loginlib.php

Language: sk (Slovak) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_sk
- **Author** Nikoleta Koncikova < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sk_manifest.php

/program/languages/sk/sk_manifest.php - description of the Slovak translation

This file defines the Slovak language package ('sk'). This file is used when this package is installed.

- **Package** waslang_sk
- **Author** Nikoleta Koncikova < translators@websiteatschool.eu>
- **Version** \$Id: sk_manifest.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/sk/was.php

Language: sk (Slovak) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_sk
- **Author** Nikoleta Koncikova < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/sk/aggregator.php

Language: sk (Slovak) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_sk
- **Author** Nikoleta Koncikova < translators@websiteatschool.eu>
- **Version** \$Id: aggregator.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/sk/htmlpage.php

Language: sk (Slovak) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_sk
- **Author** Nikoleta Koncikova < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.1 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package default Procedural Elements

ckeditor.php

- **Package** default

include_once ["ckeditor_php4.php"](#)*[line 27]*

include_once ["ckeditor_php5.php"](#)*[line 29]*

spellchecker.php

- **Package** default

void function error_handler(\$err) [*line 63*]

Function Parameters:

- **\$err**

void function escape_quote(\$str) [*line 57*]

Function Parameters:

- **\$str**

void function print_checker_results() [*line 69*]

void function print_suggs_elem(\$suggs, \$index, \$text_input_idx) [*line 43*]

Function Parameters:

- **\$suggs**
- **\$index**
- **\$text_input_idx**

void function print_textindex_decl(\$text_input_idx) [*line 31*]

Function Parameters:

- **\$text_input_idx**

void function print_textinputs_var() [*line 22*]

void function print_words_elem(\$word, \$index, \$text_input_idx) [*line 37*]

Function Parameters:

- **\$word**
- **\$index**

- **\$text_input_idx**

fckeditor.php

- **Package** default

include_once ["fckeditor_php4.php"](#)*[line 29]*

include_once ["fckeditor_php5.php"](#)*[line 31]*

fckeditor_php4.php

- **Package** default

boolean function FCKeditor_IsCompatibleBrowser() [*line 34*]

Check if browser is compatible with FCKeditor. Return true if is compatible.

crewserver.php

- **Package** default

CREW_SERVER_DATE = 2013-06-12 *[line 22]*

CREW_SERVER_NAME = CREW-server *[line 23]*

CREW_SERVER_VERSION = 0.90.5 *[line 21]*

void function agplv3_compliance() *[line 1252]*

void function dump_buffer(\$length, \$buffer, [\$message = ""]) *[line 1042]*

Function Parameters:

- **\$length**
- **\$buffer**
- **\$message**

void function get_org_property(\$origin, [\$property = 0]) *[line 1078]*

Function Parameters:

- **\$origin**
- **\$property**

void function hexdump(\$s, [\$raw = FALSE]) *[line 1089]*

Function Parameters:

- **\$s**
- **\$raw**

void function hmac(\$key, \$message, [\$raw = FALSE], [\$hash = "sha1"]) *[line 1061]*

Function Parameters:

- **\$key**
- **\$message**
- **\$raw**

- **\$hash**

void function initialise() [line 27]

void function logger(\$message, [\$level = WLOG_INFO]) [line 1010]

Function Parameters:

- **\$message**
- **\$level**

void function main() [line 60]

void function read_config([\$force = FALSE]) [line 1112]

Function Parameters:

- **\$force**

void function sockerr([\$socket = NULL]) [line 1038]

Function Parameters:

- **\$socket**

void function utf16_strlen(\$s) [line 1239]

Function Parameters:

- **\$s**

Package default Classes

Class CKEditor

[line 18]

Brief CKEditor class that can be used to create editor instances in PHP pages on server side.

- **Package** default
- **See** <c:alink:http://ckeditor.com

Sample usage:>http://ckeditor.com

Sample usage:

CKEditor::\$basePath

mixed = [line 40]

URL to the %CKEditor installation directory (absolute or relative to document root). If not set, CKEditor will try to guess it's path.

Example usage:

CKEditor::\$config

mixed = array() [line 52]

An array that holds the global %CKEditor configuration. For the list of available options,

http://docs.cksource.com/ckeditor_api/symbols/CKEDITOR.config.html

Example usage:

CKEditor::\$initialized

mixed = false [line 58]

A boolean variable indicating whether CKEditor has been initialized.

Set it to true only if you have already included `<script>` tag loading ckeditor.js in your website.

CKEditor::\$returnOutput

mixed = false [line 71]

Boolean variable indicating whether created code should be printed out or returned by a function.

Example 1: get the code creating %CKEditor instance and print it on a page with the "echo" function.

CKEditor::\$textareaAttributes

mixed = array("rows" => 8, "cols" => 60) [line 78]

An array with textarea attributes.

When %CKEditor is created with the editor() method, a HTML `<textarea>` element is created, it will be displayed to anyone with JavaScript disabled or with incompatible browser.

CKEditor::\$timestamp

mixed = D03G5XL [line 83]

A string indicating the creation date of %CKEditor.

Do not change it unless you want to force browsers to not use previously cached version of %CKEditor.

CKEditor::\$version

mixed = 3.6.6 [line 24]

The version of %CKEditor.

`\private`

CKEditor::\$_events

mixed = array() [line 88]

An array that holds event listeners.

\private

CKEditor::\$_globalEvents

mixed = array() [line 93]

An array that holds global event listeners.

\private

CKEditor::\$_timestamp

mixed = D03G5XL [line 29]

A constant string unique for each release of %CKEditor.

\private

Constructor *void* function CKEditor::CKEditor([\$basePath = null]) [line 100]

Function Parameters:

- *\$basePath* **\$basePath** (string) URL to the %CKEditor installation directory (optional).

Main Constructor.

void function CKEditor::addEventHandler(\$event, \$javascriptCode) [line 275]

Function Parameters:

- *\$event* **\$event** (string) Event name.
- *\$javascriptCode* **\$javascriptCode** (string) Javascript anonymous function or function name.

Example usage:

Adds event listener. Events are fired by %CKEditor in various situations.

void function CKEditor::addGlobalEventHandler(\$event, \$javascriptCode) [line 315]

Function Parameters:

- *\$event* **\$event** (string) Event name.
- *\$javascriptCode* **\$javascriptCode** (string) Javascript anonymous function or function name.

Example usage:

Adds global event listener.

void function CKEditor::ckeditorPath() [line 487]

Return path to ckeditor.js.

\private

void function CKEditor::clearEventHandlers([\$event = null]) [line 292]

Function Parameters:

- *\$event* **\$event** (string) Event name, if not set all event handlers will be removed (optional).

Clear registered event handlers. Note: this function will have no effect on already created editor instances.

void function CKEditor::clearGlobalEventHandlers([\$event = null]) [line 332]

Function Parameters:

- *\$event* **\$event** (string) Event name, if not set all event handlers will be removed (optional).

Clear registered global event handlers. Note: this function will have no effect if the event handler has been already printed/returned.

void function CKEditor::configSettings([\$config = array()], [\$events = array()]) [line 366]

Function Parameters:

- *\$config* **\$config** (array) The specific configurations to apply to editor instance.
- *\$events* **\$events** (array) Event listeners for editor instance.

Returns the configuration array (global and instance specific settings are merged into one array). \private

void function CKEditor::editor(\$name, [\$value =], [\$config = array()], [\$events = array()]) [line 135]

Function Parameters:

- *\$name* **\$name** (string) Name of the %CKEditor instance (this will be also the "name" attribute of textarea element).
- *\$value* **\$value** (string) Initial value (optional).
- *\$config* **\$config** (array) The specific configurations to apply to this editor instance (optional).
- *\$events* **\$events** (array) Event listeners for this editor instance (optional).

Example usage:

Creates a %CKEditor instance. In incompatible browsers %CKEditor will downgrade to plain HTML <textarea> element.

void function CKEditor::init() [line 442]

Initializes CKEditor (executed only once).

\private

string function CKEditor::jsEncode(\$val) [line 534]

Function Parameters:

- *mixed* **\$val**

This little function provides a basic JSON support. \private

void function CKEditor::replace(\$id, [\$config = array()], [\$events = array()]) [line 177]

Function Parameters:

- **\$id** **\$id** (string) The id or name of textarea element.
- **\$config** **\$config** (array) The specific configurations to apply to this editor instance (optional).
- **\$events** **\$events** (array) Event listeners for this editor instance (optional).

Example 1: adding %CKEditor to <textarea
name="article"></textarea> element:

Replaces a <textarea> with a %CKEditor instance.

void function CKEditor::replaceAll([\$className = null]) [line 220]

Function Parameters:

- **\$className** **\$className** (string) If set, replace all textareas with class className in the page.

Example 1: replace all <textarea> elements in the page.

Replace all <textarea> elements available in the document with editor instances.

void function CKEditor::returnGlobalEvents() [line 411]

Return global event handlers.

\private

void function CKEditor::script(\$js) [line 348]

Function Parameters:

- **string** **\$js**

Prints javascript code. \private

Class CrewClient

[line 363]

- **Package** default

CrewClient::\$attr

mixed = [line 381]

CrewClient::\$authenticated

mixed = FALSE [line 377]

CrewClient::\$buf_in

mixed = [line 375]

CrewClient::\$buf_out

mixed = [line 374]

CrewClient::\$cid

mixed = 0 [line 366]

CrewClient::\$date

mixed = [line 380]

CrewClient::\$headers

mixed = array() [line 372]

CrewClient::\$local_address

mixed = [line 370]

CrewClient::\$local_port

mixed = 0 [line 371]

CrewClient::\$name

mixed = [line 379]

CrewClient::\$nick

mixed = [line 378]

CrewClient::\$range

mixed = array(0, 0) [line 382]

CrewClient::\$remote_address

mixed = [line 368]

CrewClient::\$remote_port

mixed = 0 [line 369]

CrewClient::\$server

mixed = NULL [line 364]

CrewClient::\$socket

mixed = [line 365]

CrewClient::\$state

mixed = 0 [line 373]

CrewClient::\$wid

mixed = 0 [line 367]

CrewClient::\$workshop

mixed = NULL [line 376]

Constructor *void* function CrewClient::CrewClient(&\$server, \$socket, \$cid) [line 383]

Function Parameters:

- **&\$server**
- **\$socket**
- **\$cid**

void function CrewClient::frame_available(&\$buffer) [line 665]

Function Parameters:

- **&\$buffer**

void function CrewClient::frame_decode(\$data, &\$fin_opcode, &\$payload) [line 694]

Function Parameters:

- **\$data**
- **&\$fin_opcode**
- **&\$payload**

void function CrewClient::process_request(\$fin_opcode, &\$payload, &\$error) [line 444]

Function Parameters:

- **\$fin_opcode**
- **&\$payload**
- **&\$error**

void function CrewClient::recv(\$length, &\$buffer) [line 400]

Function Parameters:

- **\$length**
- **&\$buffer**

void function CrewClient::send(\$buffer) [line 659]

Function Parameters:

- **\$buffer**

void function CrewClient::valid_authentication(\$buffer) [line 481]

Function Parameters:

- **\$buffer**

void function CrewClient::valid_handshake(\$buffer) [line 528]

Function Parameters:

- **\$buffer**

Class CrewServer

[line 108]

- **Package** default

CrewServer::\$cids

mixed = 0 [line 119]

CrewServer::\$clients

mixed = array() [line 117]

CrewServer::\$dirty_sockets

mixed = array() [line 114]

CrewServer::\$mark_next

mixed = 0 [line 116]

CrewServer::\$mark_time

mixed = [line 115]

CrewServer::\$server_address

mixed = [line 109]

CrewServer::\$server_port

mixed = [line 110]

CrewServer::\$server_run_flag

mixed = FALSE [line 112]

CrewServer::\$server_socket

mixed = [line 111]

CrewServer::\$sockets

mixed = array() [line 113]

CrewServer::\$wids

mixed = 0 [*line 120*]

CrewServer::\$workshops

mixed = array() [*line 118*]

Constructor *void* function CrewServer::CrewServer([\$server_address = '0.0.0.0'], [\$server_port = 8008]) [*line 121*]

Function Parameters:

- **\$server_address**
- **\$server_port**

void function CrewServer::disconnect_client(\$socket) [*line 283*]

Function Parameters:

- **\$socket**

void function CrewServer::disconnect_socket(\$socket) [*line 272*]

Function Parameters:

- **\$socket**

void function CrewServer::dump_overview() [*line 247*]

void function CrewServer::find_workshop(\$origin, \$name, \$smax, \$wmax, \$cid, &\$error) [*line 308*]

Function Parameters:

- **\$origin**
- **\$name**
- **\$smax**
- **\$wmax**
- **\$cid**
- **&\$error**

void function CrewServer::frame_encode(\$fin_opcode, \$payload, &\$frame) [*line 343*]

Function Parameters:

- **\$fin_opcode**
- **\$payload**
- **&\$frame**

void function CrewServer::get_tv_sec() [line 223]
void function CrewServer::initialise() [line 127]
void function CrewServer::lookup_client_cid(\$socket) [line 300]
Function Parameters:

- **\$socket**

void function CrewServer::run() [line 152]

Class CrewWorkshop

[line 752]

- **Package** default

CrewWorkshop::\$attr

mixed = [line 758]

CrewWorkshop::\$clients

mixed = array() [line 753]

CrewWorkshop::\$name

mixed = [line 756]

CrewWorkshop::\$origin

mixed = [line 755]

CrewWorkshop::\$text

mixed = [line 757]

CrewWorkshop::\$wid

mixed = 0 [line 754]

Constructor *void function CrewWorkshop::CrewWorkshop(&\$server, \$wid, \$origin, \$name) [line 759]*

Function Parameters:

- **&\$server**
- **\$wid**
- **\$origin**
- **\$name**

void function CrewWorkshop::calc_jranges() [line 971]

void function CrewWorkshop::cast_message(\$client, \$message) [line 837]

Function Parameters:

- **\$client**
- **\$message**

void function CrewWorkshop::cast_userlist() [line 826]

void function CrewWorkshop::cast_user_enters(\$client) [line 829]

Function Parameters:

- **\$client**

void function CrewWorkshop::cast_user_leaves(\$client) [line 833]

Function Parameters:

- **\$client**

void function CrewWorkshop::disjoin(\$client) [line 810]

Function Parameters:

- **\$client**

void function CrewWorkshop::get_next_attr() [line 777]

void function CrewWorkshop::get_userlist() [line 766]

void function CrewWorkshop::join(&\$client, \$payload) [line 795]

Function Parameters:

- **&\$client**
- **\$payload**

void function CrewWorkshop::process_diff(&\$client, \$payload) [line 891]

Function Parameters:

- **&\$client**
- **\$payload**

void function CrewWorkshop::send(\$data, [\$encode = TRUE]) [line 877]

Function Parameters:

- **\$data**
- **\$encode**

void function CrewWorkshop::send_text_full(\$client) [line 842]

Function Parameters:

- **\$client**

void function CrewWorkshop::send_userlist(\$client) [line 870]

Function Parameters:

- **\$client**

void function CrewWorkshop::send_user_info(\$client) [line 863]

Function Parameters:

- **\$client**

Class FCKeditor

[line 74]

- **Package** default

FCKeditor::\$BasePath

string = [line 88]

Path to FCKeditor relative to the document root.

FCKeditor::\$Config

array = [line 122]

This is where additional configuration can be passed.

Example: \$oFCKeditor->Config['EnterMode'] = 'br';

FCKeditor::\$Height

mixed = [line 102]

Height of the FCKeditor. Examples: 400, 50%

FCKeditor::\$InstanceName

string = [line 82]

Name of the FCKeditor instance.

- **Access** protected

FCKeditor::\$ToolbarSet

string = [line 108]

Name of the toolbar to load.

FCKEditor::\$Value

string = [line 114]

Initial value.

FCKEditor::\$Width

mixed = [line 95]

Width of the FCKEditor. Examples: 100%, 600

Constructor *void* function FCKEditor::FCKEditor(\$instanceName) [line 130]

Function Parameters:

- *string* **\$instanceName**

Main Constructor. Refer to the `_samples/php` directory for examples.

void function FCKEditor::Create() [line 146]

Display FCKEditor.

string function FCKEditor::CreateHtml() [line 156]

Return the HTML code required to run FCKEditor.

string function FCKEditor::EncodeConfig(\$valueToEncode) [line 253]

Function Parameters:

- *string* **\$valueToEncode**

Encode characters that may break the configuration string generated by `GetConfigFieldString()`.

- **Access** protected

string function FCKeditor::GetConfigFieldString() [*line 222*]
Get settings from Config array as a single string.

- **Access** protected

boolean function FCKeditor::IsCompatible() [*line 211*]
Returns true if browser is compatible with FCKeditor.

Package waslang_sv Procedural Elements

demodata.php

/program/install/languages/sv/demodata.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.4 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/sv/install.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.4 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/sv/admin.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.5 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/sv/loginlib.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.4 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sv_manifest.php

/program/languages/sv/sv_manifest.php - description of the Swedish translation

This file defines the Swedish language package ('sv'). This file is used when this package is installed.

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: sv_manifest.php,v 1.5 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/sv/was.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.4 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/sv/aggregator.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: aggregator.php,v 1.2 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/sv/crew.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: crew.php,v 1.2 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/sv/htmlpage.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.4 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage.php

/program/modules/mailpage/languages/sv/mailpage.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: mailpage.php,v 1.2 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/sv/redirect.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.2 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/sv/sitemap.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.4 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots.php

/program/modules/snapshots/languages/sv/snapshots.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: snapshots.php,v 1.2 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/sv/axis.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.4 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/sv/cornelia.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: cornelia.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php
/program/themes/frugal/languages/sv/frugal.php
Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.4 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/sv/rosalina.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.4 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/sv/schoolyard.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.4 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php

/program/themes/sophia/languages/sv/sophia.php

Language: sv (Svenska) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_sv
- **Author** Hansje Cozijnsen < translators@websiteatschool.eu>
- **Version** \$Id: sophia.php,v 1.2 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_tr Procedural Elements

install.php

/program/install/languages/tr/install.php

Language: tr (Türkçe) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_tr
- **Author** Dirk Schouten < translators@websiteatschool.eu >
- **Version** \$Id: install.php,v 1.7 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/tr/admin.php

Language: tr (Türkçe) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_tr
- **Author** Erdil Oral < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.9 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/tr/loginlib.php

Language: tr (Türkçe) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_tr
- **Author** Özgür Gaga translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.7 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

tr_manifest.php

/program/languages/tr/tr_manifest.php - description of the Turkish translation

This file defines the Turkish language package ('tr'). This file is used when this package is installed.

- **Package** waslang_tr
- **Author** ÃœlkÃ¼ Gaga, Erdil Oral translators@websiteatschool.eu>
- **Version** \$Id: tr_manifest.php,v 1.8 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/tr/was.php

Language: tr (Türkçe) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_tr
- **Author** Erdil Oral < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.8 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/tr/axis.php

Language: tr (Türkçe) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_tr
- **Author** Erdil Oral < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.1 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/tr/cornelia.php

Language: tr (Türkçe) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_tr
- **Author** Erdil Oral < translators@websiteatschool.eu>
- **Version** \$Id: cornelia.php,v 1.1 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_ur Procedural Elements

demodata.php

/program/install/languages/ur/demodata.php

Language: ur (Ø§Ø±Ø`Ù^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.3 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/ur/install.php

Language: ur (Ø§Ø±Ø`Ù^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.3 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/ur/admin.php

Language: ur (Ø§Ø±Ø`Ù^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/ur/loginlib.php

Language: ur (Ø§Ø±Ø`Ü^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

ur_manifest.php

/program/languages/ur/ur_manifest.php - description of the Urdu translation

This file defines the Urdu language package ('ur'). This file is used when this package is installed.

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu >
- **Version** \$Id: ur_manifest.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/ur/was.php

Language: ur (Ø§Ø±Ø`Ù^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/ur/aggregator.php

Language: ur (Ø§Ø±Ø`Ù^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: aggregator.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/ur/crew.php

Language: ur (Ø§Ø±Ø`Ù^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: crew.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/ur/htmlpage.php

Language: ur (Ø§Ø±Ø`Ü^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage.php

/program/modules/mailpage/languages/ur/mailpage.php

Language: ur (Ø§Ø±Ø`Ü^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: mailpage.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/ur/redirect.php

Language: ur (Ø§Ø±Ø`Ù^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/ur/sitemap.php

Language: ur (Ø§Ø±Ø`Ù^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots.php

/program/modules/snapshots/languages/ur/snapshots.php

Language: ur (Ø§Ø±Ø`Ü^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: snapshots.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/ur/axis.php

Language: ur (Ø§Ø±Ø`Ü^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/ur/cornelia.php

Language: ur (Ø§Ø±Ø`Ù^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: cornelia.php,v 1.3 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/ur/frugal.php

Language: ur (Ø§Ø±Ø`Ù^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.3 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/ur/rosalina.php

Language: ur (Ø§Ø±Ø`Ù^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.3 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/ur/schoolyard.php

Language: ur (Ø§Ø±Ø`Ù^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.3 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php

/program/themes/sophia/languages/ur/sophia.php

Language: ur (Ø§Ø±Ø`Ù^) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_ur
- **Author** Nasira Parveen < translators@websiteatschool.eu>
- **Version** \$Id: sophia.php,v 1.3 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_vi Procedural Elements

demodata.php

/program/install/languages/vi/demodata.php

Language: vi (Tiếng Việt) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_vi
- **Author** Thao Doan < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.3 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/vi/install.php

Language: vi (Tiếng Việt) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_vi
- **Author** Thao Doan < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.4 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/vi/admin.php

Language: vi (Tiếng Việt) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_vi
- **Author** Thao Doan < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/vi/loginlib.php

Language: vi (Tiếng Việt) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_vi
- **Author** Thao Doan < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

vi_manifest.php

/program/languages/vi/vi_manifest.php - description of the Vietnamese translation

This file defines the Vietnamese language package ('vi'). This file is used when this package is installed.

- **Package** waslang_vi
- **Author** Quynh Nguyen, Thao Doan < translators@websiteatschool.eu >
- **Version** \$Id: vi_manifest.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/vi/was.php

Language: vi (Tiếng Việt) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_vi
- **Author** Thao Doan < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.3 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/vi/aggregator.php

Language: vi (Tiếng Việt) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_vi
- **Author** Quynh Nguyen < translators@websiteatschool.eu>
- **Version** \$Id: aggregator.php,v 1.2 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

althing.php

/program/modules/althing/languages/vi/althing.php

Language: vi (Tiếng Việt) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_vi
- **Author** Thao Doan < translators@websiteatschool.eu>
- **Version** \$Id: althing.php,v 1.1 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

confab.php

/program/modules/confab/languages/vi/confab.php

Language: vi (Tiếng Việt) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_vi
- **Author** Thao Doan < translators@websiteatschool.eu>
- **Version** \$Id: confab.php,v 1.1 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/vi/crew.php

Language: vi (Tiếng Việt) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_vi
- **Author** Thao Doan < translators@websiteatschool.eu>
- **Version** \$Id: crew.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/vi/htmlpage.php

Language: vi (Tiếng Việt) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_vi
- **Author** Quynh Nguyen < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.2 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage.php

/program/modules/mailpage/languages/vi/mailpage.php

Language: vi (Tiếng Việt) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_vi
- **Author** Thao Doan < translators@websiteatschool.eu>
- **Version** \$Id: mailpage.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/vi/redirect.php

Language: vi (Tiếng Việt) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_vi
- **Author** Thao Doan < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/vi/sitemap.php

Language: vi (Tiếng Việt) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_vi
- **Author** Thao Doan < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots.php

/program/modules/snapshots/languages/vi/snapshots.php

Language: vi (Tiếng Việt) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_vi
- **Author** Thao Doan < translators@websiteatschool.eu>
- **Version** \$Id: snapshots.php,v 1.3 2016/06/28 12:58:52 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/vi/axis.php

Language: vi (Tiếng Việt) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_vi
- **Author** Quynh Nguyen < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/vi/cornelia.php

Language: vi (Tiếng Việt) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_vi
- **Author** Quynh Nguyen < translators@websiteatschool.eu>
- **Version** \$Id: cornelia.php,v 1.2 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/vi/frugal.php

Language: vi (Tiếng Việt) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_vi
- **Author** Quynh Nguyen < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.2 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/vi/rosalina.php

Language: vi (Tiếng Việt) Release: 0.90.6 / 2016062900 (2016-06-29)

- **Package** waslang_vi
- **Author** Thao Doan < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.3 2016/06/28 12:58:53 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/vi/schoolyard.php

Language: vi (Tiếng Việt) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_vi
- **Author** Quynh Nguyen < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.2 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php

/program/themes/sophia/languages/vi/sophia.php

Language: vi (Tiếng Việt) Release: 0.90.5 / 2014111000 (2014-11-10)

- **Package** waslang_vi
- **Author** Quynh Nguyen < translators@websiteatschool.eu>
- **Version** \$Id: sophia.php,v 1.2 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package waslang_zh Procedural Elements

demodata.php

/program/install/languages/zh/demodata.php

Language: zh (ä, -æ-þ) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_zh
- **Author** Liu Jing Fang < translators@websiteatschool.eu>
- **Version** \$Id: demodata.php,v 1.8 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

install.php

/program/install/languages/zh/install.php

Language: zh (ä, -æ-†) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_zh
- **Author** Liu Jing Fang < translators@websiteatschool.eu>
- **Version** \$Id: install.php,v 1.9 2016/03/23 09:28:34 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

admin.php

/program/languages/zh/admin.php

Language: zh (ä, -æ-†) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_zh
- **Author** Liu Jing Fang < translators@websiteatschool.eu>
- **Version** \$Id: admin.php,v 1.10 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

loginlib.php

/program/languages/zh/loginlib.php

Language: zh (ä, -æ-†) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_zh
- **Author** Liu Jing Fang < translators@websiteatschool.eu>
- **Version** \$Id: loginlib.php,v 1.8 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

was.php

/program/languages/zh/was.php

Language: zh (ä, -æ-†) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_zh
- **Author** Liu Jing Fang < translators@websiteatschool.eu>
- **Version** \$Id: was.php,v 1.8 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

zh_manifest.php

/program/languages/zh/zh_manifest.php - description of the Chinese translation

This file defines the Chinese language package ('zh'). This file is used when this package is installed.

- **Package** waslang_zh
- **Author** Liu Jing Fang, Danny Yen < translators@websiteatschool.eu >
- **Version** \$Id: zh_manifest.php,v 1.9 2016/06/28 12:58:51 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
<info@websiteatschool.eu>
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/zh/htmlpage.php

Language: zh (ä, -æ-ţ) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_zh
- **Author** Liu Jing Fang < translators@websiteatschool.eu>
- **Version** \$Id: htmlpage.php,v 1.8 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/zh/sitemap.php

Language: zh (ä, -æ-†) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_zh
- **Author** Liu Jing Fang < translators@websiteatschool.eu>
- **Version** \$Id: sitemap.php,v 1.6 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/zh/axis.php

Language: zh (ä, -æ-†) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_zh
- **Author** Danny Yen < translators@websiteatschool.eu>
- **Version** \$Id: axis.php,v 1.7 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/zh/frugal.php

Language: zh (ä, -æ-ţ) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_zh
- **Author** Liu Jing Fang < translators@websiteatschool.eu>
- **Version** \$Id: frugal.php,v 1.8 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/zh/rosalina.php

Language: zh (ä, -æ-ţ) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_zh
- **Author** Liu Jing Fang < translators@websiteatschool.eu>
- **Version** \$Id: rosalina.php,v 1.6 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/zh/schoolyard.php

Language: zh (ä, -æ-þ) Release: 0.90.4 / 2013061400 (2013-06-14)

- **Package** waslang_zh
- **Author** Liu Jing Fang < translators@websiteatschool.eu>
- **Version** \$Id: schoolyard.php,v 1.7 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

Package wasmod_aggregator Procedural Elements

aggregator_admin.php

/program/modules/aggregator/aggregator_admin.php - management interface for aggregator-module

This file defines the administrative interface to this module. The interface consists of the following four functions.

```
aggregator_disconnect(&$output,$area_id,$node_id,$module)
aggregator_connect(&$output,$area_id,$node_id,$module)
aggregator_show_edit(&$output,$area_id,$node_id,$module,$viewonly,$edit_again,$href,$option)
aggregator_save(&$output,$area_id,$node_id,$module,$viewonly,&$edit_again,$option)
```

These functions are called from pagemanagerlib.php whenever necessary.

- **Package** wasmod_aggregator
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: aggregator_admin.php,v 1.7 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function aggregator_check_node_list(&\$item, \$area_id, \$node_id) [*line 405*]

Function Parameters:

- *array* &**\$item** holds the field definition from the \$dialogdef for the aggregator_path
- *int* **\$area_id** the area in which we are editing a snapshot module configuration
- *int* **\$node_id** the node to which the snapshot module is connected (unused)

validate and massage the user-supplied node list

this checks the node list the user entered, returns TRUE if the tests are passed. Currently the only test is checking the node numbers are ≥ 1 .

- **TODO** perhaps we should check more thoroughly for node existence but that implies also checking for user access etc. etc. We postpone that to later when the visitor's credentials will be checked against the list of nodes. So: a syntax check only. Sort of.

bool function aggregator_connect(&\$output, \$area_id, \$node_id, \$module) [line 73]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which we need to connect
- *array* **\$module** the module record straight from the database

connect this module to a node

this makes the link between the node \$node_id in area \$area_id and this module. In this case we simply link a single record to node \$node_id in a 1-to-1 relation.

Note that we set the parameters to more or less reasonable values. It is up to the user to configure the aggregator with appropriate settings.

bool function aggregator_disconnect(&\$output, \$area_id, \$node_id, \$module) [line 51]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node from which we need to disconnect
- *array* **\$module** the module record straight from the database

disconnect this module from a node

this breaks the link between the node \$node_id in area \$area_id and this module. For now we simply delete the record with the aggregator configuration data.

array function aggregator_get_dialogdef(\$viewonly) [line 269]

Function Parameters:

- *int* **\$viewonly** if TRUE the Save button is not displayed and config values cannot be changed

construct a dialog definition for the aggregator configuration data

bool function aggregator_save(&\$output, \$area_id, \$node_id, \$module, \$viewonly, &\$edit_again, \$option) [line 207]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

save the modified content data of this module linked to node \$node_id

this validates and saves the data that was submitted by the user. If validation fails, or storing the data doesn't work, the flag \$edit_again is set to TRUE and the return value is FALSE.

If the user has cancelled the operation, the flag \$edit_again is set to FALSE and the return value is also FALSE.

If the modified data is stored successfully, the return value is TRUE and the value of \$edit_again is based on the button used to save ([Save] or [Done]).

Here is a summary of return values.

\$retval | \$edit_again | Action

----- | ----- | -----

FALSE	FALSE	[Cancel] Stop editing, unlock & return to tree view
FALSE	TRUE	[Save] or [Done] failed: redo the dialog and let user correct mistakes
TRUE	FALSE	[Done] pressed, everything saved succesfully, we're done
TRUE	TRUE	[Save] pressed, everything saved succesfully but continue editing

The parameter \$option is not used in this module.

bool function aggregator_show_edit(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href, \$option) [line 133]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

present the user with a dialog to modify the aggregator that are connected to node \$node_id

this prepares a dialog for the user filled with existing data (if any), possibly allowing the user to modify the content. If the flag \$viewonly is TRUE, this routine should only display the content rather than let the user edit it. If the flag \$edit_again is TRUE, the routine should use the data available in the \$_POST array, otherwise it should read the data from the database (or wherever the data comes from). The parameter \$href is the place where the form should be POST'ed.

The dialog should be added to the \$output object. Useful routines are:

```
$output->add_content($content): add $content to the content area
$output->add_message($message): add $message to the message area (feedback to the user)
$output->add_popup_bottom($message): popup $message in browser after loading the page (uses javascript)
$output->add_popup_top($message): popup $message in browser before loading the page (uses javascript)
```

The parameter \$option is not used in this module.

aggregator_cron.php

/program/modules/aggregator/aggregator_cron.php - interface to the cron-part of the aggregator module

This file defines the interface with the aggregator-module for cron. The interface consists of this function:

```
aggregator_cron( )
```

This function is called whenever cron determines that it is time to perform this function.

- **Package** wasmod_aggregator
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: aggregator_cron.php,v 1.4 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function aggregator_cron() [*line 44*]

routine that is called periodically by cron

there is nothing in this module that needs to be done by cron, so this routine is basically a nop.

aggregator_install.php

/program/modules/aggregator/aggregator_install.php - installer of the aggregator module

This file contains the aggregator module installer. The interface consists of these functions:

```
aggregator_install(&$messages,$module_id)
aggregator_upgrade(&$messages,$module_id)
aggregator_uninstall(&$messages,$module_id)
aggregator_demodata(&$messages,$module_id,&$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wasmod_aggregator
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: aggregator_install.php,v 1.8 2016/06/16 13:48:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function aggregator_demodata(&\$messages, \$module_id, &\$config, \$manifest) [line 177]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table
- *array* **&\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy of the manifest for this module

add demonstration data to the system

this routine adds to the existing set of demonstration data as specified in `$config`.

for the sake of demonstration we add the following in the showcase section: aggregator demo-news (hidden section) news 4 news 3 news 2 news 1 we subsequently aggregate the demo-news section (which is hidden) we also try to add the snapshots to the list of nodes to aggregate, but that may prove to fail if aggregator is installed before snapshots. Mmmm, must fix that lateron.

The array \$config contains the following information.

```
$config['language_key'] => install language code (eg. 'en')
$config['dir']           => path to CMS Root Directory (eg. /home/httpd/htdocs)
$config['www']           => URL of CMS Root Directory (eg. http://exemplum.eu)
$config['progdir']       => path to program directory (eg. /home/httpd/htdocs/program)
$config['progwww']       => URL of program directory (eg. http://exemplum.eu/program)
$config['datadir']       => path to data directory (eg. /home/httpd/wasdata/alb2c3d4e5f6)
$config['user_username'] => userid of webmaster (eg. wblader)
$config['user_full_name'] => full name of webmaster (eg. Wilhelmina Bladergroen)
$config['user_email']    => email of webmaster (eg. w.bladergroen@exemplum.eu)
$config['user_id']       => numerical user_id (usually 1)
$config['demo_salt']     => password salt for all demodata accounts
$config['demo_password'] => password for all demodata accounts
$config['demo_areas']    => array with demo area data
$config['demo_groups']   => array with demo group data
$config['demo_users']    => array with demo user data
$config['demo_nodes']    => array with demo node data
$config['demo_string']   => array with demo strings from /program/install/languages/LL/demodata.php
$config['demo_replace']  => array with search/replace pairs to 'jazz up' the demo strings
```

With this information, we can add a demonstration configuration for the public area, which shows off the possibilities. Note that we add our own additions to the array \$config so other modules and themes can determine the correct status quo w.r.t. the demodata nodes etc.

bool function aggregator_install(&\$messages, \$module_id) [line 63]

Function Parameters:

- **array &\$messages** collects the (error) messages
- **int \$module_id** the key for this module in the modules table

install the module

this routine installs the module. For this module there is nothing to install, so we simply return success. The appropriate table is already created based on the tabledefs); see install/aggregator_tabledefs.php.

Note that the record for this module is already created in the modules table; the pkey is \$module_id.

bool function aggregator_uninstall(&\$messages, \$module_id) [line 121]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

uninstall the module

this is a hook for future extensions of Website@School. For now we simply return success. Any real code could look like this: DELETE FROM aggregator; to delete all existing data (but who would want that). For now this is simply a nop.

Note that bluntly deleting from the aggregator table might lead to nodes without a valid module. The better way to do it would be something like this:

```
SELECT count(node_id) AS number_of_nodes FROM aggregator;
```

```
if ($number_of_nodes > 0) then
    $messages[] = 'There are still $number_of_nodes nodes with a aggregator';
    return FALSE;
```

which in fact means that the table should already be empty before we can empty it. Oh well...

bool function aggregator_upgrade(&\$messages, \$module_id) [line 91]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

upgrade the module

this routine performs an upgrade to the installed module. Note that this 'aggregator' module currently does not need any upgrade at all because there never was an earlier version.

However, if there was to be a newer version of this module, this routine is THE place to bring the database up to date compared with the existing version. For example, if an additional field 'aggregator_extension' was to be added to the aggregator-table, it could be added using a suitable (default) value, e.g. an empty string or whatever

Any existing aggregator could then be updated here to fill the new field with data, e.g.

```
UPDATE aggregator SET aggregator_extension = "";
```

etcetera. For now this routine is a nop.

aggregator_manifest.php

/program/modules/aggregator/aggregator_manifest.php - description of the aggregator module

This file defines the various components of the aggregator module such as the names of the various scripts and version information. This file is used when this module is installed.

- **Package** wasmod_aggregator
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: aggregator_manifest.php,v 1.8 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator_search.php

/program/modules/aggregator/aggregator_search.php - interface to the search-part of this module

This file defines the interface with this module for searching content. The interface consists of a (temporary) table and this function:

```
bool aggregator_search(&$hits,&$results,$qwords, $limit, $offset)
```

The function is called whenever module content is to be searched. At that point the helper table, which is called 'search_nodes', exists and is populated.

- **Package** wasmod_aggregator
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: aggregator_search.php,v 1.6 2016/04/05 14:33:01 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool|array function aggregator_search(&\$hits, &\$results, \$qwords, \$limit, [\$offset = 0]) [line 49]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results

search this module's content in selected nodes for keywords in \$qwords

This interface is documented in [htmlpage_search\(\)](#) in [htmlpage_search.php](#).

aggregator_view.php

/program/modules/aggregator/aggregator_view.php - interface to the view-part of the module

This file defines the interface with the aggregator-module for viewing content. The interface consists of this function:

```
aggregator_view(&$output,$area_id,$node_id,$module)
```

This function is called from /index.php when the node to display is connected to this module.

- **Package** wasmod_aggregator
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: aggregator_view.php,v 1.6 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function aggregator_view(&\$theme, \$area_id, \$node_id, \$module) [*line 103*]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

display the aggregated information from the nodes linked to this aggregator

his routine grabs information from other, existing pages, and partially shows the this information followed by a 'read more...' link to the corresponding full page. Two types of pages are currently recognised:

- **htmlpage**: a configurable # of paragraphs is displayed
- **snapshots**: all images are rotated, with configurable number, time, dimensions

The following showstoppers are taken into account.

- a page must not be expired or under embargo
- the corresponding area must be accessible to the user
- database errors yield an empty list of aggregated pages
- the number of aggregated pages is limited via a config option

Specifying a section number rather than a page number is interpreted as specifying the individual pages within that section (the above showstoppers do apply, however). The combination of a limited number of pages AND the ability to specify a section id as shorthand for a collection of pages makes it easy to keep the aggregator page up-to-date with a list of the N latest pages within a section.

Note about the styling of aggregated nodes

The aggregator can be completely styled using a mix of id's and classes. Here is a rough sketch, assuming an aggregator with a non-empty header, non-empty introduction, a single snapshots node and two htmlpage modules. This all takes place within the #content div. Note that the various id's are numbered sequential within the page. Currently the DIVs .aggregator_htmlpage_clear and .aggregator_snapshots_clear are empty and unused.

```
h2 .aggregator_header
div .aggregator_introduction
div .aggregator_snapshots_outer #aggregator_outer_1
  h3 .aggregator_snapshots_header # aggregator_header_1
  div .aggregator_snapshots_inner #aggregator_inner_1
  div .aggregator_snapshots_more #aggregator_more_1
  div .aggregator_snapshots_clear #aggregator_clear_1
div .aggregator_htmlpage_outer #aggregator_outer_2
  h3 .aggregator_htmlpage_header # aggregator_header_2
  div .aggregator_htmlpage_inner #aggregator_inner_2
  div .aggregator_htmlpage_more #aggregator_more_2
  div .aggregator_htmlpage_clear #aggregator_clear_2
div .aggregator_htmlpage_outer #aggregator_outer_3
  h3 .aggregator_htmlpage_header # aggregator_header_3
  div .aggregator_htmlpage_inner #aggregator_inner_3
  div .aggregator_htmlpage_more #aggregator_more_3
  div .aggregator_htmlpage_clear #aggregator_clear_2
...
```

- **TODO** what to do with htmlpages containing h1's and h2's? Get rid of those? Mmmm....

array function aggregator_view_get_config(\$node_id) [line 153]

Function Parameters:

- *int* **\$node_id** identifies the aggregator page

retrieve the configuration information for this aggregator

array function aggregator_view_get_modules() [line 204]

retrieve a list of modules suitable for aggregation keyed by module_id

this selectively retrieves the module records for the modules we support. The information is used to determine which nodes to process and we also need a module record for the inline slideshow.

array function aggregator_view_get_nodes(&\$config, &\$tree, &\$modules) [line 229]

Function Parameters:

- *array &\$config* points to the aggregator configuration
- *array &\$tree* points to the (cached) tree of the current area
- *array &\$modules* points to array with supported modules

construct an array with the node records to aggregate

this routine converts the comma delimited list of node numbers to an array of node records, ready for further processing.

array function aggregator_view_get_node_from_db(\$node_id, &\$config, &\$modules) [line 321]

Function Parameters:

- *int \$node_id* identifies page or section to evaluate
- *array &\$config* points to the aggregator configuration
- *array &\$modules* points to array with supported modules

retrieve an array with node records straight from the database

this routine constructs a list of 0, 1 or more node records based on the \$node_id provided by the caller. The node records are retrieved from the database.

This routine takes care of the showstoppers like embargo and expiry and also access permissions to the area. We can not be sure if the user actually has access to a page until we have checked to area in which the node \$node_id resides. This is an extra test compared to above.

array function aggregator_view_get_node_from_tree(\$node_id, &\$config, &\$tree, &\$modules) [line 264]

Function Parameters:

- *int* **\$node_id** identifies page or section to evaluate
- *array* **&\$config** points to the aggregator configuration
- *array* **&\$tree** points to the (cached) tree of the current area
- *array* **&\$modules** points to array with supported modules

construct an array with node records using cached tree in current area

this routine constructs a list of 0, 1 or more node records based on the \$node_id provided by the caller. The node records are retrieved from the cached tree in &\$tree (from \$theme->tree).

This routine takes care of the showstoppers like embargo and expiry but not the area because we already have access to this area otherwise we would not be here in the aggregator module in this area.

array function aggregator_view_htmlpage(\$counter, &\$theme, &\$node, &\$config) [*line 399*]

Function Parameters:

- *int* **\$counter** is a sequential number identifying the aggregated nodes
- *object* **&\$theme** points to theme where the output goes
- *array* **&\$node** points to the node record of this htmlpage
- *array* **&\$config** points to the aggregator configuration

construct a title, summary and readmore prompt for an htmlpage page

this routine uses a heuristic approach to snip N paragraphs from the actual text in the html-page. Because we cannot be sure that stripos() is available we resort to first changing any '<P' to '<p ' and subsequently searching the string until the N+1'th '<p '. The offset we calculate this way should contain exactly N paragraphs. Obviously this assumes (dangerous...) that the htmlpage page_data actually contains paragraphs. However, if not enough '<p' strings are found, the complete page is used. Heuristics...

array function aggregator_view_snapshots(\$counter, &\$theme, &\$node, &\$config, &\$modules) [*line 471*]

Function Parameters:

- *int* **\$counter** is a sequential number identifying the aggregated nodes
- *object* **&\$theme** points to theme where the output goes
- *array* **&\$node** points to the node record of this htmlpage
- *array* **&\$config** points to the aggregator configuration
- *array* **&\$modules** points to list of records of supported modules

construct a title, inline slideshow and readmore prompt for a snapshots page

this routine uses the SnapshotViewerInline class to generate a rotating inline slideshow. This leans very heavily on JavaScript. If JavaScript is not enabled, that class has a fall-back showing the first N images statically. Graceful degradation...

aggregator_tabledefs.php

/program/modules/aggregator/install/aggregator_tabledefs.php - data definition for module

- **Package** wasmod_aggregator
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: aggregator_tabledefs.php,v 1.4 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/en/aggregator.php - translated messages for module (English)

- **Package** wasmod_aggregator
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: aggregator.php,v 1.9 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

aggregator.php

/program/modules/aggregator/languages/nl/aggregator.php - translated messages for module (Dutch)

- **Package** wasmod_aggregator
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: aggregator.php,v 1.8 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wasmod_althing Procedural Elements

althing_admin.php

/program/modules/althing/althing_admin.php - management interface for althing-module

This file defines the administrative interface to this module. The interface consists of the following four functions.

```
althing_disconnect(&$output,$area_id,$node_id,$module)
althing_connect(&$output,$area_id,$node_id,$module)
althing_show_edit(&$output,$area_id,$node_id,$module,$viewonly,$edit_again,$href,$option)
althing_save(&$output,$area_id,$node_id,$module,$viewonly,&$edit_again,$option)
```

These functions are called from pagemanagerlib.php whenever necessary.

- **Package** wasmod_althing
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: althing_admin.php,v 1.18 2016/05/26 13:22:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

TRUE function althing_config_dialog_validate(&\$dialogdef) [*line 1691*]

Function Parameters:

- *array* **&\$dialogdef** the dialog to check

validation of configuration dialog + rewriting the lists of email addresses as a side effect

bool function althing_connect(&\$output, \$area_id, \$node_id, \$module) [line 84]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which we need to connect
- *array* **\$module** the module record straight from the database

connect this module to a node

this makes the link between the node \$node_id in area \$area_id and this module. In this case we create a single 'althings' record linked to node_id via althing_id.

bool function althing_disconnect(&\$output, \$area_id, \$node_id, \$module) [line 54]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node from which we need to disconnect
- *array* **\$module** the module record straight from the database

disconnect this module from a node

this breaks the link between the node \$node_id in area \$area_id and this module. For now we simply delete the records in all tables 'althings*' using node_id == althing_id. Note the order in which the tables are emptied (to satisfy the FK constraints).

array function althing_get_dialogdef_configuration(&\$output, \$viewonly, \$node_id) [line 1298]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$viewonly** if TRUE the Save button is not displayed and config values cannot be changed
- *int* **\$node_id** identifies the current althing

construct a dialog definition for the main althing configuration

- **Uses** \$USER, - \$CFG

array function althing_get_dialogdef_moderation(&\$record, \$viewonly) [*line 1482*]

Function Parameters:

- *array* **&\$record** contains the information about this post
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed

construct the moderation dialog

array function althing_get_dialogdef_report() [*line 1705*]

construct the dialog for the reporting tool

array/bool function althing_get_post_records(\$node_id, \$sort, [\$forced = FALSE]) [*line 1631*]

Function Parameters:

- *int* **\$node_id** indicates the althing we are working with
- *int* **\$sort** is the desired sort order
- *bool* **\$forced** if TRUE the data is retrieved from DB even if it was cached

retrieve the information about all posts in an array

this routine retrieves all relevant information of all posts in this althing in a single assoc array. We do this in 1 go, which makes for a crafted piece of SQL. However, basically we are simply SELECTing all posts in this althing, in the order indicated by \$sort. As a service we also COUNT() the number of moderations a post has had AND we fetch the full name of the moderator that last changed the post.

Note that the result is cached in a local static variable, this might save some time after the user presses a button to move from post-view to list-view.

Sort order #3 is a little special: we sort on the # of moderations, allowing the user to show the

'hottest' posts re: moderation to show at the top of the list.

Sort order #4 is an attempt to order the posts by visibility. We need the CASE WHEN construct because it is hard to sort by boolean.

TRUE function althing_moderation_dialog_validate(&\$dialogdef) [line 1582]

Function Parameters:

- *array* **&\$dialogdef** the dialog to check

validation of moderation dialog

this validates the the moderation dialog. Special issue here is the contents of the remark-field: if there are no changes in the visibility and the marbles, a remark is not necessary and should not trigger an error message. We already have the limit set to 0 in the dialogdef, but we `_do_` check for the length in case of changes.

bool function althing_report_dialog_validate(&\$dialogdef, &\$posts) [line 1792]

Function Parameters:

- *array* **&\$dialogdef** to check
- *array* **&\$posts** all posts keyed by display_id (not by post_id)

validate the report selection dialog, correct the list of posts if necessary

bool function althing_save(&\$output, \$area_id, \$node_id, \$module, \$viewonly, &\$edit_again, \$option) [line 211]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$option** the selected submenu option or NULL if user used the 'Content' option to get here

dispatcher for save routines

We use different subroutines for clarity.

bool function althing_save_configuration(&\$output, \$node_id, \$viewonly, &\$edit_again, \$area_id, \$module) [line 284]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise

validate and save the modified althing configuration linked to node \$node_id

this validates and saves the data that was submitted by the user.

If validation fails, the flag \$edit_again is set to TRUE and the return value is FALSE.

If the user has cancelled the operation, the flag \$edit_again is set to FALSE and the return value is also FALSE.

If the modified data is stored successfully, the return value is TRUE. The flag edit_again is set to TRUE if the user saved via the [Save] button, and to FALSE if the user saved via the [Done] button.

bool function althing_save_moderation(&\$output, \$area_id, \$node_id, \$viewonly, &\$edit_again, \$module) [line 759]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database

- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise

validate and save the modified althing configuration linked to node \$node_id

this validates and saves the data that was submitted by the user.

If validation fails, the flag \$edit_again is set to TRUE and the return value is FALSE.

If the user has cancelled the operation, the flag \$edit_again is set to FALSE and the return value is also FALSE.

If the modified data is stored successfully, the return value is TRUE. The flag edit_again is set to TRUE if the user saved via the [Save] button, and to FALSE if the user saved via the [Done] button.

bool function althing_save_report(&\$output, \$viewonly, &\$edit_again, \$edit_again) [line 1279]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- **&\$edit_again**

a dummy routine that pretends to "save" the report selections

this is just a dummy routine to tell the caller that we want to return to [althing_show_edit_report\(\)](#) which is where the actual work is done. It is not even necessary to check for either Cancel or OK because that too is done in althing_show_edit_report(). We do set the edit_again flag.

bool function althing_save_viewlog(&\$output, \$viewonly, &\$edit_again, \$node_id) [line 1131]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which the content is connected
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed

- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise

almost a dummy routine but it must exist to handle the lone [Cancel] button

bool function althing_show_content(&\$output, \$node_id, \$module) [line 163]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

show the main content screen (in fact documentation only)

this displays the simple overview screen with documentation. It ends with a cancel button that ends the whole edit operation for this node.

bool function althing_show_edit(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href, \$option) [line 133]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *string* **\$option** the selected submenu option or NULL if user used the 'Content' option to get here

dispatcher for showing various dialog screens

there are a few dialogs that are accessed via this routine: - overview: a simple introduction to this module (via \$option==NULL) - configuration: all properties of the althing (main althings record) (\$option=configuration) - moderation: a list of postings OR an individual

post (\$option=moderation) - logmessages: a list of all moderation messages for this althing (\$option=viewlog)

This routine only handles the main overview dialog, the rest is done in subroutines.

bool function althing_show_edit_configuration(&\$output, \$node_id, \$module, \$viewonly, \$edit_again, \$href) [line 245]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

show a simple edit screen for the althing configuration

bool function althing_show_edit_moderation(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href) [line 414]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

present the user with a dialog for moderation of althing posts at node \$node_id

this is a complex routine that is responsible for constructing and showing the dialogs associated with posts.

The following dialogs exist here. #1. Posts overview

Id	Date	E	V	M	Name	Subject
d1	yyyy-mm-dd hh:mm	1	V	0	Nickname	Subject
d2	yyyy-mm-dd hh:mm	0	V	0	Nickname	Subject
d3	yyyy-mm-dd hh:mm	1	H	0	Nickname	Subject
d4	yyyy-mm-dd hh:mm	0	V	0	Nickname	Subject
d5	yyyy-mm-dd hh:mm	0	V	0	Nickname	Subject
d6	yyyy-mm-dd hh:mm	0	U	0	Nickname	Subject

[Cancel]

E=Number of edits, V=Visibility:Visible/Hidden/Unpublished, M=Marbles

#2. Moderation of an individual posting

Prev Up Next

ID: dd (ppp)
Date: yyyy-mm-dd hh:mm:ss
From: Nickname <email@address> (registered)
Subject: Subject
IP-address: 127.0.0.1
Xx xx xx xxxxx xx xxxx xx xxxxx xx xxxx

Status: () Unpublished (X) Published
Visible: () Hidden (X) Visible
Marbles: _____
Remarks:

[Save] [Done] [Cancel]

Prev Up Next

There are some complex interactions between these screens and the corresponding save routine [althing_save_moderation\(\)](#). The effects we try to achieve are as follows.

The listing in dialog #1 shows all posts. Every post ID is a link to dialog #2. The [Cancel] button in dialog #1 ends the editing of this node; ie. returns to Page Manager.

Dialog #2 can be used to change the visibility of a post or to modify the number of marbles. The contents of the remarks field eventually ends up in the log file. The [Cancel] button here bring the user back to dialog #1. In effect this is a kind of zoom function: you can zoom in on a post and go back to the list without leaving this node, etc. etc. Note that the link Up also takes you back to the list in dialog #1, be it that in that case we use afragment to arrive at the correct position in the list. The [Done] button saves the modified post (and the remarks) and

also return to dialog #1. The [Save] button always has the meaning "save the data and continue editing".

Note that saving an unchanged record does not yield a message in the log because that would only add noise.

As a service there are simple links Prev, Up and Next which allow for quickly stepping through the posts in detail without having to go back to the list in dialog #1.

bool function althing_show_edit_moderation_list(&\$output, &\$records, \$node_id, \$sort_index, \$href) [line 472]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *array* **&\$records** is the ordered array with all posts in this althing
- *int* **\$node_id** the node to which this module is connected
- *int* **\$sort_index** indicates the currently selected sort order
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

show a tabular overview of all posts in this althing

this shows the main moderation screen: a list of all posts in a user-defined sort order. Re-ordering can be done by clicking the column header. Click twice to reverse the order. Due to space constraints the contents of the moderations column, visibility column and marbles column are just a single character. The explanation of these ad hoc codes is given in the title. Note that the name column also shows the email address via a mouse over (again, to save valuable space).

The list ends with a [Cancel] button which ends the complete edit of this node (i.e. returns to Page Manager).

bool function althing_show_edit_moderation_post(&\$output, &\$record, \$node_id, \$sort_index, \$viewonly, \$edit_again, &\$records) [line 600]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *array* **&\$records** is the ordered array with all posts in this althing
- *int* **\$node_id** the node to which this module is connected
- *int* **\$sort_index** indicates the currently selected sort order

- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- **&\$record**

show a single post and allow for moderation of this post

this shows the current contents of the post and allows the moderator to change the visibility and the number of marbles assigned to this post. The moderator is encouraged to add a remark to every moderation action.

void function althing_show_edit_report(&\$output, \$node_id, \$module, \$edit_again, \$href) [line 1147]

Function Parameters:

- **&\$output**
- **\$node_id**
- **\$module**
- **\$edit_again**
- **\$href**

bool function althing_show_edit_viewlog(&\$output, \$node_id, \$module, \$href) [line 980]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

show a list of moderation messages to the user

This is basically a dump of the althing_log_messages table. However, we have tried to jazz it up a little.

The features are as follows. You can arrive here via de link in the submenu. In that case we simply show the available records (all available records) in ascending order, grouped by post_id. This dump consists of a single row in the HTML-table with essential information

about the posting to which the messages belong, That row is followed by two rows per logmessage, containinf the date of moderation, the moderator name, the remark and the action.

An additional feature is the link in the header line (the 1st column: ID): clicking that link reverses the order of the list. Also every posting has their own link (via the ID). This link leads to the moderation screen (dialog #2) where this posting can be edited. In that individual post dialog there is a link back to here. However, in this case the output of this routine is limited to the selected posting.

There is also a link in the moderation list (in the moderations column) which yields the messages for a single posting.

The layout of the table is more or less as follows:

ID	Date	Name	Subject/remark
10	yyyy-mm-dd hh:mm	Anonymous Coward	4 Re: Chocolate galore
	yyyy-mm-dd hh:mm	Maria Montessori	Remark of Maria
here-comes-the-action-text-spanning-three-columns			

require_once **dirname(__FILE__)."../althing_common.php"** [*line 39*]

althing_common.php

/program/modules/althing/althing_common.php - code shared between admin and view

This file defines various constants and subroutines used from both and [althing_view.php](#).

- **Package** wasmod_althing
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: althing_common.php,v 1.7 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
ALTHING_DEFAULT_VISIBILITY = ALTHING_DEFAULT_VISIBILITY_HIDDEN [line 38]
ALTHING_DEFAULT_VISIBILITY_HIDDEN = 0 [line 34]
ALTHING_DEFAULT_VISIBILITY_REGISTERED = 1 [line 35]
ALTHING_DEFAULT_VISIBILITY_VISIBLE = 2 [line 36]
ALTHING_HARVEST_LIMIT = 10 [line 39]
ALTHING_MARBLES_LIMIT = 10 [line 40]
ALTHING_STATUS_CLOSED = 2 [line 32]
ALTHING_STATUS_FROZEN = 1 [line 31]
ALTHING_STATUS_OPENED = 0 [line 30]
array function althing_get_emails($buffer) [line 58]
```

Function Parameters:

- *string* **\$buffer** holds the messy list of email addresses

create a neat array with email-addresses from a (possibly messy) text

this routine converts a (messy) list of email addresses to an array with a single address per array element. If there are no addresses, an empty array is returned. The addresses are supposed to be comma-delimited and/or newline and/or space delimited. Note that we expect only plain email addresses, no readable names etc. (KISS)

```
bool function althing_send_alerts($emails, $moderator, $private, $visible, $area_id, $post) [line 113]
```

Function Parameters:

- *array* **\$emails** holds all email addresses to send an alert to

- **bool \$moderator** TRUE if emails are of moderators, FALSE means subscribers
- **bool \$private** TRUE if the althing lives in a private area
- **bool \$visible** TRUE if post already published, FALSE if still unpublished
- **int \$area_id** holds the page with this althing
- **array \$post** the information about the post

send an alert message to every email address in \$emails

this routine sends an individual message to each of the addresses in the array \$emails notifying the receiver about the new post. Relevant information (headers) are taken from the \$post-record. Note that we attempt to leak as less information as is possible, eg. we don't show email addresses of posters, or the contents, or other information like the name of the page (which could be in a private area) or the title of the althing.

There are several combinations flags that yield different mails to be sent. Here is a truth-table:

Visible	Moderator	Private	Message	Subject
FALSE	FALSE	FALSE	public_published	[nn] Public post #xx is now published
FALSE	FALSE	TRUE	private_published	[nn] Private post #xx is now published
FALSE	TRUE	FALSE	moderation	[nn] Moderation request for post #xx
FALSE	TRUE	TRUE	moderation	[nn] Moderation request for post #xx
TRUE	FALSE	FALSE	public_visible	[nn] New public post #xx
TRUE	FALSE	TRUE	private_visible	[nn] New private post #xx
TRUE	TRUE	FALSE	public_visible	[nn] New public post #xx
TRUE	TRUE	TRUE	private_visible	[nn] New private post #xx

Note that most messages contain a warning about the need to login first (either for accessing the private area OR for accessing the admin interface for moderation). In total there are 5 different messages. These can be adapted via the Translate Tool.

Note that the messages 'public_published' and 'private_published' are to be sent once a moderator publishes an unpublished post (ie. makes the post visible for the first time). This means that in that case we are called from althing_admin.php at moderation time rather than from althing_view.php at submit-time.

Note: the URL in the mail now uses the new parameter 'post' to select a single message for display (see also [althing_show_overview\(\)](#) in althing_view.php). By showing only a single post, the user is not distracted by other messages that were already there (July 2014).

string function bbcode2html(\$bbcode, [\$q = FALSE]) *[line 232]*

Function Parameters:

- **string \$bbcode** the BBCode-string to translate
- **bool \$q** TRUE means any URLs are fully qualified, FALSE uses the short variety

convert valid BBCode to valid HTML

this routine more or less replaces BBCode tags with the HTML equivalent. We assume that the BBCode is validated before. If you are unsure, you can use [bbcode_validate\(\)](#) to double check (at a cost of processing time).

Strategy here is to

- get rid of any superfluous CR's and CR+LF's
- defuse any HTML-tricks via htmlspecialchars() (note that this also converts the " in [url="path"...[/url])
- convert the 'simple' tags via a cheap routine str_replace()
- convert the tags with parameters (path, name) via the more expensive preg_replace

Since we probably need to use this routine over and over again in a single call, we buffer the preg search and replace arrays on the first call, speeding up subsequent calls, at a cost of a single if (is_null()). The plain search and replace arrays are so static that we simply declare them static from the start.

Note 1: We do get rid of a single newline after [quote] or [quote=...] because that would otherwise ruin the display with an extra blank line in the quote div. If you really want to have a blank line, use two newlines in a row.

Note 2: We do take into consideration the friendly URL feature and also an option for fully qualified domain names.

- Uses [bbcode_callback_q\(\)](#)
- Uses [bbcode_callback\(\)](#)

string function bbcode_callback(\$m, [\$q = FALSE]) [line 283]

Function Parameters:

- *array* **\$m** holds complete matched code + all individual sub-matches
- *bool* **\$q** TRUE means any URLs are fully qualified, FALSE uses the short variety

handle advanced BBCode search/replace specifically for encoding filenames

we use this callback routine to perform the regex-based search/replace so we get a chance to encode the URLs and also deal with the proc-y-friendly style of servind files via file.php.

Note that we are transparent about 'hidden' URLs: if the user uses the format [url=path]anchor[/url] we show the path in plain text too, like so: anchor (anchor) No surprises with deceiving links..

- Usedby [bbcode_callback_q\(\)](#)
- Usedby [bbcode2html\(\)](#)

string function `bbcode_callback_q($m)` [line 265]

Function Parameters:

- *array* **\$m** holds complete matched code + all individual sub-matches

one-line wrapper to allow for fully qualified URLs

- Usedby [bbcode2html\(\)](#)
- Uses [bbcode_callback_q\(\)](#)

void function `bbcode_help(&$theme)` [line 569]

Function Parameters:

- *object* **&\$theme** collects (html) output

show a table illustrating the possible BBCode

this routine is a help function. If JS is enabled, the information is hidden and the user can make it visible by clicking on the title.

bool function `bbcode_validate(&$item)` [line 344]

Function Parameters:

- *array* **&\$item** an element of a dialogdef that supposedly holds BBCode

validate the BBCode in the dialogdef item

* Simple tags always should match in pairs, without overlap: [b] [/b] [u] [/u] [i] [/i] [s] [/s] [code] [/code]

The more complicated constructs with opening/closing tags are: [url=path]anchor[/url]
[url]path[/url] [img]path[/img] [quote]text[/quote] [quote=author]text[/quote]
[quote="author"]text[/quote]

There is one single tag, ie without closing tag:

[img=path]

Path should be a path relative to the data root, e.g. '/areas/exemplum/snapshots/cyanara.jpg'. This limits the URLs and the IMGs to files on this server.

Note that we also validate the specified path via which also checks permissions of the current user. This prevents leaking of files via pupils from the teachers private intranet.

bool/string function bbcode_valid_path(\$filename) [line 501]

Function Parameters:

- *string* **\$filename** the file to check, relative to CFG->datadir

validate a filename for existence and access

this routine checks to see if the current user is allowed to access the file \$filename and also that it actually exists. We return FALSE if access is denied or the cleaned up path if all is well.

althing_cron.php

/program/modules/althing/althing_cron.php - interface to the cron-part of the althing module

This file defines the interface with the althing-module for cron. The interface consists of this function:

```
althing_cron()
```

This function is called whenever cron determines that it is time to perform this function.

- **Package** wasmod_althing
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: althing_cron.php,v 1.3 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function althing_cron() [*line 44*]

routine that is called periodically by cron

there is nothing in this module that needs to be done by cron, so this routine is basically a nop.

althing_install.php

/program/modules/althing/althing_install.php - installer of the althing module

This file contains the althing module installer. The interface consists of these functions:

```
althing_install(&$messages,$module_id)
althing_upgrade(&$messages,$module_id)
althing_uninstall(&$messages,$module_id)
althing_demodata(&$messages,$module_id,&$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wasmod_althing
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: althing_install.php,v 1.16 2016/04/05 08:52:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function althing_demodata(&\$messages, \$module_id, &\$config, \$manifest) [*line 147*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table
- *array* **&\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this module

add demonstration data to the system

this routine adds demodata for this module to the database.

Note that a routine by this name should exist (even if it does nothing).

Note If the module is installed via the Install Wizard, this routine is called. However, if a module is installed as an additional module after installation, the `{ $module }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

The array `$config` contains the following information.

```
$config['language_key'] => install language code (eg. 'en')
$config['dir']           => path to CMS Root Directory (eg. /home/httpd/htdocs)
$config['www']           => URL of CMS Root Directory (eg. http://exemplum.eu)
$config['progdir']       => path to program directory (eg. /home/httpd/htdocs/program)
$config['progwww']       => URL of program directory (eg. http://exemplum.eu/program)
$config['datadir']       => path to data directory (eg. /home/httpd/wasdata/alb2c3d4e5f6)
$config['user_username'] => userid of webmaster (eg. wblader)
$config['user_full_name'] => full name of webmaster (eg. Wilhelmina Bladergroen)
$config['user_email']    => email of webmaster (eg. w.bladergroen@exemplum.eu)
$config['user_id']       => numerical user_id (usually 1)
$config['demo_salt']     => password salt for all demodata accounts
$config['demo_password'] => password for all demodata accounts
$config['demo_areas']    => array with demo area data
$config['demo_groups']   => array with demo group data
$config['demo_users']    => array with demo user data
$config['demo_nodes']    => array with demo node data
$config['demo_string']   => array with demo strings from /program/install/languages/LL/demodata.php
$config['demo_replace']  => array with search/replace pairs to 'jazz up' the demo strings
```

With this information, we can add a demonstration configuration for the public area, which shows off the possibilities. Note that we add our own additions to the array `$config` so other modules and themes can determine the correct status quo w.r.t. the demodata nodes etc.

bool function althing_install(&\$messages, \$module_id) [line 61]

Function Parameters:

- *array &\$messages* collects the (error) messages
- *int \$module_id* the key for this module in the modules table

install the module

this routine installs the module. For this module there is nothing to install, so we simply return success.

Note that the record for this module is already created in the modules table; the pkey is `$module_id`.

bool function althing_uninstall(&\$messages, \$module_id) [line 92]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

uninstall the module

this is a hook for future extensions of Website@School. For now we simply return success.

bool function althing_upgrade(&\$messages, \$module_id) [*line 78*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

upgrade the module

this routine performs an upgrade to the installed module. Note that the althing module does not need any upgrade at all because there never was an earlier version.

For now this routine is a nop.

althing_manifest.php

/program/modules/althing/althing_manifest.php - description of the althing module

This file defines the various components of the althing module such as the names of the various scripts and version information. This file is used when this module is installed.

- **Package** wasmod_althing
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: althing_manifest.php,v 1.6 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

althing_search.php

/program/modules/althing/althing_search.php - interface to the search-part of this module

This file defines the interface with this module for searching content. The interface consists of a (temporary) table and this function:

```
bool althing_search(&$hits,&$results,$qwords, $limit, $offset)
```

The function is called whenever module content is to be searched. At that point the helper table, which is called 'search_nodes', exists and is populated.

- **Package** wasmod_althing
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: althing_search.php,v 1.4 2016/04/05 09:25:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool|array function althing_search(&\$hits, &\$results, \$qwords, \$limit, [\$offset = 0]) [line 64]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results

search this module's content in selected nodes for keywords in \$qwords

This interface is documented in [htmlpage_search\(\)](#) in [htmlpage_search.php](#).

We have to search the following fields in two tables:

a. althings: 1 record/althing: header/introduction/question/conclusion/footer

b. althing_posts: 0,1,...N records/althing: subject/author/content

Althings always qualify (the 5 fields are always visible) Posts only qualify when althing status is 0 (open) or 1 (frozen) AND the post is published and visible

Hits in althings link to the main page (node=\$node_id) Hits in the posts link to node=\$node_id,post=\$post_id

The content in posts is converted from BBCode to HTML before starting the search,

bool/int function althing_search_results(\$table, \$where) [*line 190*]

Function Parameters:

- *string* **\$table** name of table to search
- *string* **\$where** where clause to select hits in \$table

count the number of hits in \$table

require_once **dirname(__FILE__)." /althing_common.php"** [*line 37*]

althing_view.php

/program/modules/althing/althing_view.php - interface to the view-part of the althing module

This file defines the interface with the althing-module for viewing content. The interface consists of this function:

```
althing_view(&$output,$area_id,$node_id,$module)
```

This function is called from /index.php when the node to display is connected to this module.

- **Package** wasmod_althing
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: althing_view.php,v 1.23 2016/05/26 13:22:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
ALTHING_DIRNAME = dirname(__FILE__) [line 37]
```

```
ALTHING_REFERENCE = sha1(__FILE__."":__LINE__) [line 40]
```

```
void function althing_show_form(&$theme, $config, $dialogdef) [line 868]
```

Function Parameters:

- *object* **&\$theme** collects (html) output
- *array* **\$config** holds the althing configuration parameters
- *array* **\$dialogdef** defines the various fields in the dialog

show the form where the user/visitor can compose a new post

this routine displays the form where a registered user or an anonymous visitor can compose a new post. There is a difference between the two groups: - users cannot change their name + email; this is copied from \$USER, - visitors are allowed to pick their own nicknames and email address Another difference is that only registered users (with sufficient permissions) have an additional link that leads to the file browser.

The file browser allows for inserting a full path into the message area, at the current cursor position. This way the user does not have to type the filename/path when inserting a URL or

an IMG.

bool function althing_show_overview(&\$theme, \$config) [line 225]

Function Parameters:

- *object* **&\$theme** collects (html) output
- *array* **\$config** holds the althing configuration parameters

display the althing information and all published posts OR a single post

this is the main screen that users/visitors usually see on arrival: the header/intro/question, a list of 0,1 or more published posts, and the conclusion and the footer. There are also buttons to [Post] a message or [Harvest] exiting messages.

However, this routine also takes care of requests for a single, individual post. This post is specified via the command line parameter 'post'. If it is specified AND if it identifies a valid (published) message within this Althing that single post is displayed, together with an OK-button. This function is used to direct subscribers and moderators to a specific single message. Note that most links from the single message (parents, children and even self) bring the visitor to the whole althing, ie. they then leave the single post view. (July 2014)

bool function althing_show_preview(&\$theme, \$config, \$dialogdef, \$ip_addr, \$parents, &\$dialogdef) [line 959]

Function Parameters:

- *object* **&\$theme** collects (html) output
- *array* **\$config** althing configuration record straight from althings table
- *array* **&\$dialogdef** defines the dialog and will hold POSTed values
- *string* **\$ip_addr** is the address the user/visitor is calling from
- *array* **\$parents** contains parent_id's of this post
- **\$dialogdef**

show a preview of the post to the user, including references to parents etc.

the user can either submit the post or go back to edit the post

bool function althing_show_thankyou(&\$theme, \$config, \$ip_addr, \$post, &\$post) [line 1189]

Function Parameters:

- *object* **&\$theme** collects (html) output
- *array* **\$config** althing configuration record straight from althings table
- *string* **\$ip_addr** is the address the user/visitor is calling from
- *array* **&\$post** a self-contained array returning all information about the post
- **\$post**

display a thank-you message and the URL of the newly added post

bool function althing_submit_message(&\$theme, \$config, \$data, \$ip_addr, \$delay, &\$post) [line 1040]

Function Parameters:

- *object* **&\$theme** collects (html) output
- *array* **\$config** althing configuration record straight from althings table
- *array* **\$data** contains the (validated) dialogdef and also the post's parents if any
- *string* **\$ip_addr** is the address the user/visitor is calling from
- *int* **\$delay** the # of seconds since time=t0
- *array* **&\$post** a self-contained array returning all information about the post

store the new post, add relations and send mail to moderators (and maybe subscribers too)

this routine takes care of storing the (validated) data in `$data['dialogdef']` and `$data['parents']` into the various tables in the database. If all goes well, we also send a mail to the moderators and, if the post was published immediately, the subscribers too. On success, the array `$post` is filled with the post data, including the newly assigned `post_id`. It is up to the caller to do something with this returned post data, e.g. display it or whatever.

Storing the data takes a few separate steps. First we need to calculate a `display_id` which is unique within this althing. Once we have that, we can store the actual post, which yields a fresh (unique) `post_id`. Subsequently we can use that `post_id` to store the parent-child-relations in the database.

If that all works out, we always inform the moderators and, if the post was published, also the subscribers. If the post is not visible, the subscribers will be sent a mail lateron, if and when the post is published by a moderator.

bool function althing_view(&\$theme, \$area_id, \$node_id, \$module) [line 52]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

display the content of the althing linked to node \$node_id

bool function althing_view_dialog_validate(&\$dialogdef) [line 923]

Function Parameters:

- *array* **&\$dialogdef** defines the dialog and will hold POSTed values

validate the data entered by the user/visitor

Other than the standard validation we check for at least 1 @-sign in the email address. (It is non-trivial to validate an email address, it may not even be possible to do it in a single regular expression. See <http://stackoverflow.com/questions/201323> for more information.) Note that we only check this field if it is not viewonly. Reason: the user is not able to edit a viewonly field, so an error there would be a fatal error. We don't want that.

Also, we validate the BBCode in the message.

array function althing_view_get_dialogdef(\$config, \$token_key, [\$posts = NULL]) [line 628]

Function Parameters:

- *array* **\$config** althing configuration record straight from althings table
- *string* **\$token_key**
- **\$posts**

construct a dialog definition for the add a post form

This defines the add a post form.

bool function althing_view_get_posts(&\$theme, \$althing_id, &\$posts, [\$sort = 1], [\$show = 1]) [line 559]

Function Parameters:

- *object* **&\$theme** collects (html) output
- *int* **\$althing_id** aka \$node_id
- *array* **&\$posts** receives all posts (could be empty array)
- *int* **\$sort** is the sort order of the posts for display: 1=ascending, -1=descending
- *int* **\$show** indicates actual visibility of redacted posts: 1=show redacted, 0=hide redacted.

get all posts for this althing into an array for easy output

this retrieves data from the database in a specific order (if necessary). via \$show=0 it is possible to actually suppress the output of the redacted posts

bool function althing_view_show_post(&\$theme, &\$post, [\$visible = TRUE], [\$frozen = FALSE], [\$preview = TRUE], [\$odd_even = 'odd'], [\$href = '']) [line 423]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **&\$post** a self-contained array with all information about a single post
- *bool* **\$visible** TRUE is show post, FALSE is show redacted post
- *bool* **\$frozen** FALSE indicates if the althing is open for replies, TRUE means althing is frozen
- *bool* **\$preview** TRUE indicates whether we show this post in preview mode, FALSE=not
- *string* **\$odd_even** used to link CSS parameters to odd and even posts
- *string* **\$href** links to this althing page, used to create absolute links to relations

output a single post via \$theme

[] \$author \$datim (#\$msgid) Subject: \$subject Marbles: \$marbles (reply to \$msgid) In reply to: \$p1 Replies: \$c1, \$c2, \$c3 Harvests: \$h1

bbcode2html(Content)

Note "In reply to: \$p1" might also be 'Harvested from: \$p2, \$p3, \$p4' depending

on the post.

Note: preview mode means that the displayed post does not have links to elsewhere which would let the user leave the current page. This is handy for showing a preview of a message that is not yet posted.

Note: the parameter href (default "") is used in the single post display to make sure that the links to the "relations" (parent, children) are absolute, i.e. they link to the full althing page, including headers etc. etc. (the single post page shows only a single post, without the context of the althing).

bool|null|array function check_add_reply_harvest(&\$theme, \$althing_id, \$status) [*line 768*]

Function Parameters:

- *object* **&\$theme** collects (html) output
- *int* **\$althing_id** aka \$node_id
- *int* **\$status** tells us about Open/Frozen/Closed status of the althing

check for user action indicating wish to add a post, reply to a post or harvest posts

this checks to see if the user has taken action to start adding a new post. This is the case if

1. the user pressed the [Post] button
2. the user clicks the reply-link in an existing post's header
3. the user selected 0, 1 or more existing posts and pressed the [Harvest] button

If we find that the user does want to add/reply/harvest, we collect the 0, 1 or more posts in an array, keyed by post_id. This array is then returned. If the user does NOT want to add/reply/harvest, we return NULL. If the althing is currently closed or frozen (or rather not open), we tell the user about that and the return value is FALSE.

Note: In step 3A below we identify the checked posts by looking at the \$_POST[] array. This is a shortcut, because usually we first setup a dialogdef and subsequently check all the elements in the \$dialogdef for their values. We now sort of trust the data in the POST-array; an ordinary user will only be able to have just the checkboxes from the overview checked, leading to an element with key 'p' followed by the post_id (in decimal). An attacker might try to fool us by crafting a post with post_id's from another althing, eg. one hidden in a protected area. We mitigate that by explicitly requiring the althing_id in the WHERE-clause we will be constructing. Another possible danger is specifying many, many, many post_id's. We will limit that via the limit of posts that can be harvested in an althing (there must be a limit, somewhere, right?).

`require_once ALTHING_DIRNAME."/althing_common.php" [line 38]`

`require_once $CFG->progdir."/lib/tokenlib.php" [line 41]`

althing_tabledefs.php

/program/modules/althing/install/althing_tabledefs.php - data definition for module

A total of 4 tables are defined:

althings - 1 record per althing, tied to a node; contains main configuration
althing_posts - 0, 1 or more records containing visitor comments to this althing
althing_relations - 0, 1 or more records store the parent-child relations between posts
althing_log_messages - 0, 1 or more moderator messages/log messages

The tables 'althings' and 'althing_posts' and have straightforward 1-to-N relations. The 'althing_relations' table stores the connections between messages. This requires a separate table because of the special feature of an althing that allows for a reply to more than one post at the same time. We need to keep track of the multiple parents in a convenient way. The table 'althing_log_messages' keeps track of all moderation events with structured descriptions and optional moderator remarks.

- **Package** wasmod_althing
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: althing_tabledefs.php,v 1.6 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

althing.php

/program/modules/althing/languages/en/althing.php - translated messages for module (English)

- **Package** wasmod_althing
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: althing.php,v 1.38 2016/05/26 13:22:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

althing.php

/program/modules/althing/languages/nl/althing.php - translated messages for module (Dutch)

- **Package** wasmod_althing
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: althing.php,v 1.10 2016/05/26 13:22:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wasmod_confab Procedural Elements

confab_admin.php

/program/modules/confab/confab_admin.php - management interface for confab-module

This file defines the administrative interface to this module. The interface consists of the following four functions.

```
confab_disconnect(&$output,$area_id,$node_id,$module)
confab_connect(&$output,$area_id,$node_id,$module)
confab_show_edit(&$output,$area_id,$node_id,$module,$viewonly,$edit_again,$href,$option)
confab_save(&$output,$area_id,$node_id,$module,$viewonly,&$edit_again,$option)
```

These functions are called from pagemanagerlib.php whenever necessary.

- **Package** wasmod_confab
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: confab_admin.php,v 1.16 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

TRUE function confab_config_dialog_validate(&\$dialogdef) [line 1145]

Function Parameters:

- *array* **&\$dialogdef** the dialog to check

validation of configuration dialog + rewriting passcode (no trailing/leading spaces)

bool function confab_connect(&\$output, \$area_id, \$node_id, \$module) [*line 85*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which we need to connect
- *array* **\$module** the module record straight from the database

connect this module to a node

this makes the link between the node \$node_id in area \$area_id and this module. In this case we create a single 'confabs' record linked to node_id via confab_id.

- **Uses \$USER**

bool function confab_disconnect(&\$output, \$area_id, \$node_id, \$module) [*line 55*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node from which we need to disconnect
- *array* **\$module** the module record straight from the database

disconnect this module from a node

this breaks the link between the node \$node_id in area \$area_id and this module. For now we simply delete the records in all tables relating to 'confabs*' using node_id == confab_id. Note the order in which the tables are emptied (to satisfy the FK constraints).

array function confab_get_conversations(\$confab_id) [*line 1354*]

Function Parameters:

- *int* **\$confab_id**

retrieve all conversations in reverse chronological order

if something goes wrong, we return an empty array but we do write to the logger

array function confab_get_dialogdef_configuration(&\$output, \$viewonly, \$node_id) [line 934]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$viewonly** if TRUE the Save button is not displayed and config values cannot be changed
- *int* **\$node_id** identifies the current confab

construct a dialog definition for the main confab configuration

array function confab_get_dialogdef_moderation(&\$participants, \$viewonly) [line 1158]

Function Parameters:

- *array* **&\$participants** contains all participants in this conversation
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed

construct the moderation dialog

array function confab_get_dialogdef_report(&\$conversations) [line 1221]

Function Parameters:

- *array* **&\$conversations** ordered list of conversations holds data for list box

construct the dialog for the reporting tool

bool function confab_save(&\$output, \$area_id, \$node_id, \$module, \$viewonly, &\$edit_again, \$option) [line 209]

Function Parameters:

- *object* **&\$output** collects the html output (if any)

- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$option** the selected submenu option or NULL if user used the 'Content' option to get here

dispatcher for save routines

We use different subroutines for clarity.

bool function confab_save_configuration(&\$output, \$node_id, \$viewonly, &\$edit_again) [line 277]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which the content is connected
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise

validate and save the modified confab configuration linked to node \$node_id

this validates and saves the data that was submitted by the user.

If validation fails, the flag \$edit_again is set to TRUE and the return value is FALSE.

If the user has cancelled the operation, the flag \$edit_again is set to FALSE and the return value is also FALSE.

If the modified data is stored successfully, the return value is TRUE. The flag edit_again is set to TRUE if the user saved via the [Save] button, and to FALSE if the user saved via the [Done] button.

- **Uses \$USER**

bool function confab_save_moderation(&\$output, \$viewonly, &\$edit_again, \$edit_again) [line 676]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- **&\$edit_again**

a dummy routine that pretends to "save" the moderation selections

this is just a dummy routine to tell the caller that we want to return to [confab_show_edit_moderation\(\)](#) which is where the actual work is done. It is not even necessary to check for either Cancel or OK because that too is done in [confab_show_edit_report\(\)](#). We do set the edit_again flag.

bool function confab_save_report(&\$output, \$viewonly, &\$edit_again, \$edit_again) [line 917]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- **&\$edit_again**

a dummy routine that pretends to "save" the report selections

this is just a dummy routine to tell the caller that we want to return to [confab_show_edit_report\(\)](#) which is where the actual work is done. It is not even necessary to check for either Cancel or OK because that too is done in [confab_show_edit_report\(\)](#). We do set the edit_again flag.

bool function confab_show_content(&\$output, \$node_id, \$module) [line 161]

Function Parameters:

- *object* **&\$output** collects the html output (if any)

- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

show the main content screen (in fact documentation only)

this displays the simple overview screen with documentation. It ends with a cancel button that ends the whole edit operation for this node.

bool function confab_show_edit(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href, \$option)
[line 136]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *string* **\$option** the selected submenu option or NULL if user used the 'Content' option to get here

dispatcher for showing various dialog screens

there are a few dialogs that are accessed via this routine: - overview: a simple introduction to this module (via \$option==NULL) - configuration: all properties of the confab (main confabs record) (\$option=configuration) - moderation: show current with option to PM users or close the conversation (\$option=moderation) - report: a list of _all_ messages in a conversation in this confab (\$option=report)

bool function confab_show_edit_configuration(&\$output, \$node_id, \$module, \$viewonly, \$edit_again, \$href) [line 240]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected

- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

show a simple edit screen for the confab configuration

bool function confab_show_edit_moderation(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href) [line 404]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

present the user with one of two dialogs for moderation of confab posts at node \$node_id

the moderation functionality is split into two different dialogs.

The Messagelist dialog looks like this:

```
[Refresh] [Intervene] [Cancel]
Message N
Message N-1
...
Message 1
```

The Refresh button updates the list with messages. The Intervene button leads to the second dialog (see below). The Cancel button returns to the overarching Content dialog.

The Intervention-dialog Show a dialog to the user (the moderator) where she can break into the conversation and send (personal) messages to participants OR remove a participant OR

remove `_all_` participants from the conversation (or end the conversation).

The dialog looks a bit like this:

Action

() ~Send a message

() ~Remove from conversation

Participant

{listbox-with-participants}

~Message:

[_____]

[~OK] [~Cancel]

(list of `_all_` messages in reverse order (latest at the top))

The OK button performs the selected action (~Send or ~Remove) with the select Participant (or All participants). If the Message input is not empty, that message is added to the conversation.

The possible scenarios are as follows: - Send the message in 'Message:' to All (using [OK]) - Send the message in 'Message:' to the participant selected in {listbox} (using [OK]) - Remove the selected participant from the conversation (using [OK]) - Effectively end the conversation by removing All participants (using [OK]) - [Cancel] takes the moderator back to the Messagelist dialog screen

The list of messages is ordered in reverse chronological order. The moderator sees `_all_` messages, including personal messages The moderator is NOT visible for the regular participants; the name does not show up in the userlist

A moderator can send a personal message to a participant; the participant cannot talk back other than by using a message to all. The moderation only works on the current conversation. Removing all participants from the conversation bumps the conversation too

- **TODO** this routine is too long

bool function confab_show_edit_moderation_list(&\$output, \$confab_id, \$conversation_id, \$viewonly, \$href) [line 577]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$confab_id**

- *int* **\$conversation_id**
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

show a button bar and list of messages in the selected conversation

bool function confab_show_edit_moderation_messages(&\$output, \$confab_id, \$conversation_id) [line 614]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$confab_id**
- *int* **\$conversation_id**

create a list of all message in chronological order descending

bool function confab_show_edit_report(&\$output, \$node_id, \$module, \$edit_again, \$href) [line 725]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

present the user with a dialog for creating reports on confab conversations at node \$node_id

This is basically the following dialog followed by the results of the constructed query based on the previous dialog contents:

Con~versation

[listbox with conversations]

Fields:

☐ ~Message ID
☐ ~Author
☐ ~Remote address
☐ ~Date
☐ ~Text

Messages from:

☐ ~System
☐ ~User
☐ ~Both

Messages to:

☐ ~Group
☐ ~Individual
☐ Bot~h

[~OK] [~Cancel]

The fields checkboxes determine which data is displayed. The two radio buttons are added to the selection criteria (AND'ed together). If the selections yield no data, message is displayed, otherwise all checked fields of the selected records are displayed.

`require_once dirname(__FILE__)."/confab_common.php" [line 39]`

confab_common.php

/program/modules/confab/confab_common.php - code shared between admin and view

This file defines various constants and subroutines used from both and [confab_view.php](#).

- **Package** wasmod_confab
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: confab_common.php,v 1.8 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
CONFAB_ACTION_MESSAGE = 1 [line 55]
CONFAB_ACTION_REMOVE = 2 [line 56]
CONFAB_ANONYMOUS_POSTFIX = * [line 53]
CONFAB_BRAILLE = 0 [line 42]
CONFAB_MAX_CONVERSATION_LIMIT = 86400 [line 49]
CONFAB_MAX_PARTICIPANTS_LIMIT = 100 [line 50]
CONFAB_MAX_TRIES_LIMIT = 25 [line 51]
CONFAB_PERSONAL_NO = 0 [line 34]
CONFAB_PERSONAL_REGISTERED = 1 [line 35]
CONFAB_PERSONAL_YES = 2 [line 36]
CONFAB_REFRESH_LIMIT = 2 [line 52]
CONFAB_STATUS_CLOSED = 0 [line 30]
CONFAB_STATUS_OPENED = 2 [line 32]
CONFAB_STATUS_REGISTERED = 1 [line 31]
CONFAB_USER_STATUS_JOINED = 1 [line 39]
CONFAB_USER_STATUS_LEAVING = 2 [line 40]
CONFAB_USER_STATUS_NONE = 0 [line 38]
CONFAB_VISUAL = 1 [line 43]
CONFAB_VISUAL_BW = 2 [line 44]
CONFAB_VISUAL_BY = 4 [line 46]
CONFAB_VISUAL_RB = 3 [line 45]
bool function confab_conversation_timeout(&$config, [$bump_participant = FALSE]) [line 133]
```

Function Parameters:

- **array &\$config** holds the confab configuration record
- **bool \$bump_participant** if TRUE a new participant_id is made available

check conversations for timeout and hard limit and maybe create a new participant_id

this is a dual-purpose routine. The first task is to check the validity of the current

conversation. If the conversation started longer than CONFAB_MAX_CONVERSATION_LIMIT (24h) ago, we bluntly end the conversation by incrementing the last_conversation_id in the confab configuration. The same effect happens when the latest contribution to the conversation was more than \$timeout_conversation (1h) ago.

If the conversation is no longer current, the last_conversation_id is incremented and also the last_participant_id is reset to 0. This information will be recorded in the database and also updated in the \$config array (which was provided by reference, and not by value).

However... in order to save a trip to the database, we can also provide a new participant_id in the same go. This happens when \$bump_participant is TRUE. The caller can be sure that in that case the value \$config['last_participant_id'] can be used by the caller (provided the routine returns TRUE, indicating success).

If somehow something goes wrong, e.g. a database error, the \$config record is untouched and the routine returns FALSE.

This routine should be called periodically, for instance - when a new participants wants to join a conversation - when a participant contributes to the conversation This keeps the conversation_id in line and makes sure that an ongoing conversation is stopped eventually.

array function confab_get_participants(&\$config) [*line 69*]

Function Parameters:

- *array* **&\$config** holds the confab configuration

retrieve all participants that joined the current conversation

string function confab_remote_addr() [*line 92*]

shorthand for remote IP-address

confab_cron.php

/program/modules/confab/confab_cron.php - interface to the cron-part of the confab module

This file defines the interface with the confab-module for cron. The interface consists of this function:

```
confab_cron( )
```

This function is called whenever cron determines that it is time to perform this function.

- **Package** wasmod_confab
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: confab_cron.php,v 1.3 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function confab_cron() [*line 44*]

routine that is called periodically by cron

there is nothing in this module that needs to be done by cron, so this routine is basically a nop.

confab_install.php

/program/modules/confab/confab_install.php - installer of the confab module

This file contains the confab module installer. The interface consists of these functions:

```
confab_install(&$messages,$module_id)
confab_upgrade(&$messages,$module_id)
confab_uninstall(&$messages,$module_id)
confab_demodata(&$messages,$module_id,&$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wasmod_confab
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: confab_install.php,v 1.8 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function confab_demodata(&\$messages, \$module_id, &\$config, \$manifest) [line 147]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table
- *array* **&\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this module

add demonstration data to the system

this routine adds demodata for this module to the database.

Note that a routine by this name should exist (even if it does nothing).

Note If the module is installed via the Install Wizard, this routine is called. However, if a module is installed as an additional module after installation, the `{ $module }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

The array `$config` contains the following information.

```
$config['language_key'] => install language code (eg. 'en')
$config['dir']          => path to CMS Root Directory (eg. /home/httpd/htdocs)
$config['www']          => URL of CMS Root Directory (eg. http://exemplum.eu)
$config['progdir']      => path to program directory (eg. /home/httpd/htdocs/program)
$config['progwww']      => URL of program directory (eg. http://exemplum.eu/program)
$config['datadir']      => path to data directory (eg. /home/httpd/wasdata/alb2c3d4e5f6)
$config['user_username'] => userid of webmaster (eg. wblader)
$config['user_full_name'] => full name of webmaster (eg. Wilhelmina Bladergroen)
$config['user_email']   => email of webmaster (eg. w.bladergroen@exemplum.eu)
$config['user_id']      => numerical user_id (usually 1)
$config['demo_salt']    => password salt for all demodata accounts
$config['demo_password'] => password for all demodata accounts
$config['demo_areas']   => array with demo area data
$config['demo_groups']  => array with demo group data
$config['demo_users']   => array with demo user data
$config['demo_nodes']   => array with demo node data
$config['demo_string']  => array with demo strings from /program/install/languages/LL/demodata.php
$config['demo_replace'] => array with search/replace pairs to 'jazz up' the demo strings
```

With this information, we can add a demonstration configuration for the public area, which shows off the possibilities. Note that we add our own additions to the array `$config` so other modules and themes can determine the correct status quo w.r.t. the demodata nodes etc.

bool function confab_install(&\$messages, \$module_id) [line 61]

Function Parameters:

- *array &\$messages* collects the (error) messages
- *int \$module_id* the key for this module in the modules table

install the module

this routine installs the module. For this module there is nothing to install, so we simply return success.

Note that the record for this module is already created in the modules table; the pkey is `$module_id`.

bool function confab_uninstall(&\$messages, \$module_id) [line 92]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

uninstall the module

this is a hook for future extensions of Website@School. For now we simply return success.

bool function confab_upgrade(&\$messages, \$module_id) [*line 78*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

upgrade the module

this routine performs an upgrade to the installed module. Note that the confab module does not need any upgrade at all because there never was an earlier version.

For now this routine is a nop.

confab_manifest.php

/program/modules/confab/confab_manifest.php - description of the confab module

This file defines the various components of the confab module such as the names of the various scripts and version information. This file is used when this module is installed.

- **Package** wasmod_confab
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: confab_manifest.php,v 1.4 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

confab_search.php

/program/modules/confab/confab_search.php - interface to the search-part of this module

This file defines the interface with this module for searching content. The interface consists of a (temporary) table and this function:

```
bool confab_search(&$hits,&$results,$qwords, $limit, $offset)
```

The function is called whenever module content is to be searched. At that point the helper table, which is called 'search_nodes', exists and is populated.

- **Package** wasmod_confab
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: confab_search.php,v 1.4 2016/04/05 14:33:01 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool|array function confab_search(&\$hits, &\$results, \$qwords, \$limit, [\$offset = 0]) [line 55]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results

search this module's content in selected nodes for keywords in \$qwords

This interface is documented in [htmlpage_search\(\)](#) in [htmlpage_search.php](#).

this module requires an extra where clause to select only those confabs that are open for registered users (status = 1) or open for anyone (status = 2). this does NOT mean that conversations can be searched (they can't) but either way the header/introduction/footer can

be searched. Only if status=0 there cannot be any results because there is just a message 'this confab closed'.

confab_view.php

/program/modules/confab/confab_view.php - interface to the view-part of the confab module

This file defines the interface with the confab-module for viewing content. The interface consists of this function:

```
confab_view(&$output,$area_id,$node_id,$module)
```

This function is called from /index.php when the node to display is connected to this module.

- **Package** wasmod_confab
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: confab_view.php,v 1.24 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
CONFAB_DIRNAME = dirname(__FILE__) [line 37]
```

```
CONFAB_REFERENCE = sha1(__FILE__."": "__LINE__") [line 40]
```

```
bool function confab_braille_dispatcher(&$theme, &$config, &$participant) [line 882]
```

Function Parameters:

- *object* **&\$theme** collects the output
- *array* **&\$config** holds the confab configuration record
- *array* **&\$participant** holds the participant record

dispatcher for braille interface

this routine decides what needs to be done (and does it). As a side effect, we check various time-out conditions.

1. we ALWAYS check for timeout of (other) participants, this may or may not change the status of others from active to idle
2. we ALWAYS check if we are kicked out, e.g. because the conversation timed out, we were directed to leave via a

status change (maybe initiated by the moderator) or the Confab was suddenly closed (by a moderator). In this case we silently leave the conversation and perform the Quit action which will return us to the regular \$theme.

3. we SOMETIMES check for conversations that time out, i.e. whenever a contribution is posted (to the current conversation), we check and maybe increment the conversation_id. This means that we effectively kick ourselves out the next time we come here.

Note that most of the subroutines called handle their own I/O and set the \$silent_mode of the \$theme to TRUE, effectively suppressing the regular theme. This leaves a blank screen to 'play' with. The notable exception is [confab_leave_conversation\(\)](#): that routine DOES rely on \$theme to build a regular page including navigation, etc.

string function confab_braille_page_headers() [line 1229]

send various headers attempting to defeat caching

this is an attempt to force the client not to reuse the data that will be sent shortly. As a bonus, we also construct a string with the corresponding http-equiv statements, to further convince the browser not to cache the page. Reference:
<http://www.htmlgoodies.com/beyond/reference/article.php/3472881>
<http://stackoverflow.com/questions/49547>

While we are here, we add a few more useful headers such as Content-Type.

bool function confab_braille_show_all(&\$theme, &\$config, &\$participant) [line 1110]

Function Parameters:

- *object* **&\$theme** collects the output
- *array* **&\$config** holds the confab configuration record
- *array* **&\$participant** holds the participant record

send all messages to the participant; update last_seen_id

this not just shows the new messages, but it shows _all_ message.

- **TODO** this looks a lot like the [confab_braille_show_main\(\)](#), maybe there is room for improvement.

bool function confab_braille_show_main(&\$theme, &\$config, &\$participant) [line 939]

Function Parameters:

- *object* **&\$theme** collects the output
- *array* **&\$config** holds the confab configuration record
- *array* **&\$participant** holds the participant record

construct the main/messages screen for the braille interface

this routine checks if there are any new messages for \$participant. If so, these are displayed and a basic bare bones button bar is displayed. Here the user can acknowledge the messages (by pressing [OK]), acknowledge the messages and immediately start typing a new message (by pressing [Talk]) or move to the display of all messages that were exchanged since the conversation starten (by pressing [All]).

If there are no new messages a single 'Talk' button (with an asterisk rather than the word 'Talk') is displayed. This page automatically refreshes itself, making sure that the server is regularly polled for any new messages.

bool function confab_braille_show_talk(&\$theme, &\$config, &\$participant) [line 1026]

Function Parameters:

- *object* **&\$theme** collects the output
- *array* **&\$config** holds the confab configuration record
- *array* **&\$participant** holds the participant record

show a dialog where a user can write a message or can navigate to another confab-screen

this routine sends a dialog to the user that looks a bit like this: Message _____ to <listbox> [Send] [Cancel] [All] [Users] [Quit]

In this dialog the user can enter a message, select a destination from the <listbox> (if enabled) and submit the message via [Send]. Alternatively the user can cancel this screen (and return to the main/messages screen) or navigate to one of the other screens [All] or [Users] or simply end the conversation via [Quit].

bool function confab_braille_show_users(&\$theme, &\$config, &\$participant) [line 1168]

Function Parameters:

- *object* **&\$theme** collects the output
- *array* **&\$config** holds the confab configuration record
- *array* **&\$participant** holds the participant record

send a list of users to the participant

bool function confab_join(&\$theme, &\$config) [line 146]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **&\$config** holds the confab configuration including the conversation_id

handle the complete procedure to join the conversation

this routine handles joining the conversation. We use the token lib, to force participants to first load an empty join dialog before entering credentials and gaining access.

the container \$data holds the number of attempts to specify the correct passcode. If this exceeds the maximum, we leave the user with an error in an otherwise empty screen. They can try again after that, but they have to start with a fresh token.

bool function confab_join_conversation(&\$theme, &\$config, &\$dialogdef) [line 516]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **&\$config** holds the confab configuration including the conversation_id
- *array* **&\$dialogdef** defines the dialog and will hold POSTed values

do housekeeping and setup the opening screen with requested interface

this routine is called after the user authenticated successfully. The credentials are in \$dialogdef and we are quite confident that the data was valid only moments ago and probably still is.

we now need to add the collected data to the confab_participants table. as a side effect we may have to bump the last_conversation_id ie if the current conversation has ended (after a

time out), we start a new one. In that case we are the first/only participant so any name clash we may have (due to race condition) is mitigated.

bool function confab_join_dialog_show(&\$theme, &\$config, &\$dialogdef) [line 245]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **&\$config** holds the confab configuration including the conversation_id
- *array* **&\$dialogdef** defines the dialog and will hold POSTed values

output the join dialog including the form and header/introduction/footer

this routine can yield a dialog that includes fields for both the user's username and password (as used in a system login). If this is the case (indicated by the presence of the field 'login_password'), we add the paramter 'login=1' to our own URL in order to make a stop at the system login procedure (in) before proceeding with our own join dialog (killing two birds with one stone).

bool function confab_join_dialog_validate(&\$dialogdef, &\$config) [line 297]

Function Parameters:

- *array* **&\$dialogdef** defines the dialog and will hold POSTed values
- *array* **&\$config** holds the Confab configuration

validate the data entered by the user/visitor

Other than the standard validation we check the passcode and for anonymous visitors we make sure the username is unique within the conversation.

Note 1: We distinguish between Anonymous users and Registered users by adding a postfix to the nickname of the anonymous user. This postfix makes it possible to choose a name like an existing name, eg. 'hparkh' without making it impossible for the real 'hparkh' to login using her real username. As a result, there could be two participants, one named 'hparkh' and one named (say) 'hparkh*'. For the name lookup below these names are different.

Note 2: There is a possible race condition here if two participants use the same nickname at the same time. However, the database will enforce that only one of the two will 'win' and be entered into the confab_participants table. The other one will get a key violation error and, sadly, has to try again. IMHO an acceptable risk but YMMV.

- **TODO** We may have to enforce usernames cannot end in an asterisk in Account Manager.

array function confab_join_get_dialogdef(\$config, \$token_key) [line 394]

Function Parameters:

- array **\$config** confab configuration record
- string **\$token_key**

construct a dialog definition for the join screen

this defines the join form. There are several flavours, depending on the confab configuration and the fact that the user is (already) logged in or not.

The basic join screen looks a bit like this: Nickname:

Passcode:

Interface:

<listbox>

[OK]

If a user is currently logged in, we can already enter the username. If above that the confab is of type registered users only, we make the name readonly, since the user is not allowed to join anonymously. (She must present the passcode though).

If the user is not logged in, we cannot already pre-load the username. If above that the confab is of type registered users only, we add the username and password fields from the system login so we can login and join in a single go. The reason to take this approach is that we don't want to bother the user with two screens: one with username/password and another with username passcode. IF the user provides the correct password AND passcode, we're in business. If she misses one of the two, we either 'hang' in the system login, OR in a loop for requesting a passcode from an already logged in user.

The difference between the dialogs can be detected from the existence of either the 'username' OR the 'login_username' entry in \$dialogdef. Alternatively, the input field 'login_password' is only added if a username/password authentication is required too.

Note: The length of the nickname/username in the Confab tables is set at 64. A system username is limited to a length of 60. This allows for at most 4 characters to distinguish Anonymous and Registered participants in the participants and messages tables. Feels like a

kludge. Sorry.

bool function confab_leave_conversation(&\$theme, &\$config, &\$participant, [\$reason = ""]) [line 758]

Function Parameters:

- *object* **&\$theme** collects the output
- *array* **&\$config** holds the confab configuration record
- *array* **&\$participant** holds the participant record
- *string* **\$reason** indicates why we are leaving (or we were made to)

end the conversation for \$participant (at their own request or forced)

this is the opposite of the [confab_join_conversation\(\)](#) routine: we destroy the cookie and remove the participants record etc. The \$reason gives the user feedback as to what just happened: if the system has closed down the conversation due to a time-out or when the moderator has closed the Confab altogether, the \$reason explains. If no specific reason is given, we show a default message.

bool function confab_save_message(&\$theme, &\$config, &\$participant) [line 664]

Function Parameters:

- *object* **&\$theme** collects the output
- *array* **&\$config** holds the confab configuration record
- *array* **&\$participant** holds the participant record

add non-empty message to database

Note that there is a chance that we post this message to conversation N (as recorded in our participant-record) whereas the confab has moved on to conversation N+1. Therefore we do hold on to the conversation_id in the participants record rather than the last_conversation_id in the config record. Once this message is processed, the participant will find out that the conversation has ended and that she lands on the login screen...

bool function confab_view(&\$theme, \$area_id, \$node_id, \$module) [line 66]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

display the content of the confab linked to node \$node_id

This routine is called whenever a visitor visits node \$node_id. It depends on the joined-status what is actually sent to the browser. If the participant already joined the conversation, indicated by presenting a valid cookie, the request is further handled in the corresponding protocol dispatcher.

If not, this must be another participant attempting to join the conversation. If the Confab is not closed AND the maximum number of simultaneous participants is not yet reached, we enter the procedure to let the visitor present her credentials (passcode) etc. This is done in a separate routine. This keeps this routine short and readable.

int function confab_view_calc_participants(&\$config) [*line 1697*]

Function Parameters:

- *array* **&\$config** holds the confab configuration including the conversation_id

calculate the number of currently joined participants in the current conversation

this calculates the number of participants in the current conversation in the confab. We only count those that have joined the conversation (status is CONFAB_USER_STATUS_JOINED) and not those that are already marked as leaving the conversation (status is CONFAB_USER_STATUS_LEAVING) or are otherwise not joined (status is CONFAB_USER_STATUS_NONE).

In case of error, we return the maximum number of participants. This may be the wrong answer but at least we won't be letting too many participants in.

bool/int function confab_view_delete_stale_participants(&\$config) [*line 1821*]

Function Parameters:

- *array* **&\$config** holds the confab configuration record

check for stale participants and remove corresponding records

this routine keeps the database clean; any participants that were last seen more than CONFAB_MAX_CONVERSATION_LIMIT (24h) ago. As long as this routine is called periodically, say every time a participant joins a conversation we are able to weed out all the garbage, eventually.

array function confab_view_get_messages(&\$config, \$nickname, [\$last_message_id = 0]) [*line 1736*]

Function Parameters:

- *array* **&\$config** holds the confab configuration
- *string* **\$nickname** for who are we retrieving the messages
- *int* **\$last_message_id** the last message \$nickname has already received

retrieve all messages in the conversation since \$last_message_id

this routine returns an array with all messages with an id that is greater than \$last_message_id, ie. the messages that were added since \$last_message_id was recorded. If \$last_message_id = 0, _all_ messages in the conversation are retrieved.

Note that this routine also takes the 'personal messages' into account: if a message is personal, it is only added to the result if \$nickname is either the sender or the receiver of the personal message. Others are not allowed to see these personal messages.

The result is always an array (could be empty) and it is always keyed by confab_message_id.

bool/int function confab_view_participants_timeout(&\$config) [*line 1778*]

Function Parameters:

- *array* **&\$config** holds the confab configuration record

check for idle participants and timed out participants

we update all participants for which the status is active but that posted a message longer than timeout_idle ago. If there are any participants with a changes status, we flag that to all participants by setting the show_list flag. That means that the next time a participant returns a fresh userlist has to be sent to the browser (if applicable).

This routine also flags users that have passed the conversation timeout value, i.e. that have not touched home for a long time. Those participants are more or less kicked out, eventually freeing space for other participants.

bool function confab_visual_dispatcher(&\$theme, &\$config, &\$participant) [line 1294]

Function Parameters:

- *object* **&\$theme** collects the output
- *array* **&\$config** holds the confab configuration record
- *array* **&\$participant** holds the participant record

dispatcher for visual interface

this routine decides what needs to be done (and does it). As a side effect, we check various time-out conditions.

1. we ALWAYS check for timeout of (other) participants, this may or may not change the status of others from active to idle
2. we ALWAYS check if we are kicked out, e.g. because the conversation timed out, we were directed to leave via a status change (maybe initiated by the moderator) or the Confab was suddenly closed (by a moderator). In this case we
 - a. either silently leave the conversation and perform the Quit action which will return us to the regular \$theme, OR
 - b. send a Q-instruction to the client, requesting the client to quit the proper way.
3. we SOMETIMES check for conversations that time out, i.e. whenever a contribution is posted (to the current conversation), we check and maybe increment the conversation_id. This means that we effectively kick ourselves out the next time we come here.

Note that some of the subroutines called handle their own I/O and set the \$silent_mode of the \$theme to TRUE, effectively suppressing the regular theme. This leaves a blank screen or an open connection to 'play' with. The notable exception is : that routine DOES rely on \$theme to build a regular page including navigation, etc.

Note that this routine handles both asynchronous requests (indicated by having 'button_send' or 'button_update' set in \$_POST) which take place behind the scenes or requests that involve a complete HTML-page (e.g. after the user presses [Refresh]).

The way to quit the conversation is to kindly ask the client to submit a Quit-request rather than simply showing the quit-screen: the latter would probable confuse the asynchronous handler in the client.

bool function confab_visual_show_main(&\$theme, &\$config, &\$participant) [line 1357]

Function Parameters:

- *object* **&\$theme** collects the output
- *array* **&\$config** holds the confab configuration record
- *array* **&\$participant** holds the participant record

construct the main screen for the Visual interfae

this sets up the initial page that will be controlled via Javascript lateron (including resizing).

We generate the page based on the information that is currently available, i.e. the current list of (all) messages, the current list of (all) participants and an up to date listbox (if applicable).

This same routine is called whenever the user presses the Refresh-button, This is a way to refrek both lists if somehow something went wrong underway.

bool function confab_visual_update(&\$theme, &\$config, &\$participant, [\$force_quit = FALSE]) *[line 1600]*

Function Parameters:

- *object* **&\$theme** collects the output
- *array* **&\$config** holds the confab configuration record
- *array* **&\$participant** holds the participant record
- **\$force_quit**

send an update to the client in resonse to an XMLHttpRequest

Note that we cannot process TAB ('\t') and LF ('\n') in JavaScript because these delimiters are used in our protocol. We have to get rid of those.

This routine conditionally sends updates to the client. The following messages are defined:

Send a message "M" | datim | nickfrom | nickto | message

Send a complete userlist "U" | n | m | nick1 | name1 | id1 | active1 ...
| nickn | namen | idn | activen

Nicely ask the client to quit the conversation: "Q"

Indicate end of file: "Z"

The client is supposed not to process any lines following the "Z" command (which is always sent). That allows us to add a minimal escape to the main page via URL, for clients

that have trouble with XMLHttpRequest functionality. Even though it doesn't work the client is able to get to a more readable page...

```
require_once CONFAB_DIRNAME."/confab_common.php" [line 38]
```

```
require_once $CFG->progdire."/lib/tokenlib.php" [line 41]
```

confab_tabledefs.php

/program/modules/confab/install/confab_tabledefs.php - data definition for module

A total of 3 tables is defined:

confabs - 1 record per confab, tied to a node; contains main configuration
confab_messages - 0, 1 or more records containing visitor contributions to conversations
confab_participants - 0, 1 or more records containing transient visitor information

The tables 'confabs' and 'confab_messages' have a relative straightforward 1-to-N relation. A record in confab_participants is created once a visitor joins the conversation (and presenting the correct pass code). The record only exists while the visitor is engaged in the conversation. Once a visitor leaves the conversation, the corresponding record is deleted. A conversation is completely self-contained in the confab_messages table.

Note: The length of the username in the table confab_participants below is set at 64. A system username is limited to a length of 60. This allows for at most 4 characters to distinguish Anonymous and Registered participants in the participants and messages tables. Feels like a kludge. Sorry. But wait, it gets worse... Because a registered user cannot change her username on joining the conversation, we have to have a way to distinguish multiple instances of the same USER->username. The best I could think of was to add a numeric 'username extension' that counts from 1 to N. Also, I now make a distinction between the username, ie. what the user entered in the dialog and the nickname which is used in the messages table. For anonymous users the nickname is the username with a postfix. For registered users it is their USER->username, with a postfix '(nnn)' for the second and following instances. Ugly. The extra space is 12: enough for '(' and 10 digits and ')'. A positive 32-bit number fits in that space...

- **Package** wasmod_confab
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: confab_tabledefs.php,v 1.11 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

confab.php

/program/modules/confab/languages/en/confab.php - translated messages for module (English)

- **Package** wasmod_confab
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: confab.php,v 1.28 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

confab.php

/program/modules/confab/languages/nl/confab.php - translated messages for module (Dutch)

- **Package** wasmod_confab
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: confab.php,v 1.7 2016/03/23 09:28:21 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wasmod_crew Procedural Elements

crew_admin.php

/program/modules/crew/crew_admin.php - management interface for crew-module

This file defines the administrative interface to this module. The interface consists of the following four functions.

```
crew_disconnect(&$output,$area_id,$node_id,$module)
crew_connect(&$output,$area_id,$node_id,$module)
crew_show_edit(&$output,$area_id,$node_id,$module,$viewonly,$edit_again,$href,$option)
crew_save(&$output,$area_id,$node_id,$module,$viewonly,&$edit_again,$option)
```

These functions are called from pagemanagerlib.php whenever necessary.

- **Package** wasmod_crew
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: crew_admin.php,v 1.6 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
CREW_ACL_ROLE_READONLY = CREW_PERMISSION_READ [line 41]
CREW_ACL_ROLE_READWRITE = CREW_PERMISSION_READ|CREW_PERMISSION_WRITE [line 42]
CREW_PERMISSION_READ = 1 [line 39]
CREW_PERMISSION_WRITE = 2 [line 40]
bool function crew_connect(&$output,$area_id,$node_id,$module) [line 89]
```

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides

- *int* **\$node_id** the node to which we need to connect
- *array* **\$module** the module record straight from the database

connect this module to a node

this makes the link between the node \$node_id in area \$area_id and this module. In this case we simply link a single workshops record to node \$node_id in a 1-to-1 relation. Any permissions (ACL) can be stored in the acls_modules_node table later but by default we start afresh with just a bare workshops record.

Note that we set the parameter 'visibility' to 0. This implies a workshop that is only visible for the individual accounts that are explicitly allowed to view and/or edit. It is up to the user to configure the workshop in a different way, e.g allow all accounts to view the workshop results (visibility=1) or allow the world to see the results (visibility=2).

bool function crew_disconnect(&\$output, \$area_id, \$node_id, \$module) [*line 58*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node from which we need to disconnect
- *array* **\$module** the module record straight from the database

disconnect this module from a node

this breaks the link between the node \$node_id in area \$area_id and this module. For now we simply delete the relevant record from the workshops table. Also we remove any record which might exist in the ACL table acls_modules_nodes for this particular node.

array function crew_get_dialogdef(&\$output, \$viewonly, \$module_id, \$area_id, \$node_id, \$user_id) [*line 350*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$viewonly** if TRUE the Save button is not displayed and values cannot be changed
- *int* **\$module_id** indicates the id of the crew module in the database (needed for ACL)
- *int* **\$area_id** indicates the area where node_id lives (needed for ACL)

- *int* **\$node_id** indicates which page we are looking at (needed for ACL)
- *int* **\$user_id** indicates the current user (needed for ACL)

construct a dialog definition for the workshop configuration

this generates an array which defines the dialog for workshop configuration. There are a few plain fields that simply go into the appropriate workshops record and the save and cancel button. However, there may also be items related to ACLs. These fields are used to define the user's roles and the should be presented in a table. We abuse the field names for this purpose: if the first 3 characters are 'acl' we put the widgets in an HTML-table, otherwise it is just an ordinary widget.

Note that in the case of a single simple user without any acquaintances (ie. a user that is not member of any group) the user is not able to add herself to the list of authorised users. What is the point of having a `_collaborative_` workshop when you are the only one to collaborate with? (However, it would be fairly easy to force/add an entry for this user if the tmp table would turn out empty. Maybe later....?)

bool function `crew_save(&$output, $area_id, $node_id, $module, $viewonly, &$edit_again, $option)` [line 248]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

save the modified content data of this module linked to node \$node_id

this validates and saves the data that was submitted by the user. If validation fails, or storing the data doesn't work, the flag `$edit_again` is set to TRUE and the return value is FALSE.

If the user has cancelled the operation, the flag `$edit_again` is set to FALSE and the return value is also FALSE.

If the modified data is stored successfully, the return value is TRUE and the value of \$edit_again is based on the button used to save ([Save] or [Done]).

Here is a summary of return values.

\$retval	\$edit_again	Action
-----	-----	-----
FALSE	FALSE	[Cancel] Stop editing, unlock & return to tree view
FALSE	TRUE	[Save] or [Done] failed: redo the dialog and let user correct mistakes
TRUE	FALSE	[Done] pressed, everything saved succesfully, we're done
TRUE	TRUE	[Save] pressed, everything saved succesfully but continue editing

The parameter \$option is not used in this module.

bool function crew_show_edit(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href, \$option) [*line 143*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

present the user with a dialog to modify the workshop that is connected to node \$node_id

this prepares a dialog for the user filled with existing data (if any), possibly allowing the user to modify the content. If the flag \$viewonly is TRUE, this routine should only display the content rather than let the user edit it. If the flag \$edit_again is TRUE, the routine should use the data available in the \$_POST array, otherwise it should read the data from the database (or wherever the data comes from). The parameter \$href is the place where the form should be POST'ed.

The dialog should be added to the \$output object. Useful routines are:

```
$output->add_content($content): add $content to the content area
$output->add_message($message): add $message to the message area (feedback to the user)
$output->add_popup_bottom($message): popup $message in browser after loading the page (uses JS)
$output->add_popup_top($message): popup $message in browser before loading the page (uses JS)
```

The parameter \$option is not used in this module.

crew_cron.php

/program/modules/crew/crew_cron.php - interface to the cron-part of the workshop (CREW) module

This file defines the interface with the module for cron. The interface consists of this function:

```
crew_cron()
```

This function is called whenever cron determines that it is time to perform this function.

- **Package** wasmod_crew
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: crew_cron.php,v 1.4 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function crew_cron() [*line 44*]

routine that is called periodically by cron

there is nothing in this module that needs to be done by cron, so this routine is basically a nop.

crew_install.php

/program/modules/crew/crew_install.php - installer of the crew module

This file contains the crew module installer. The interface consists of these functions:

```
crew_install(&$messages,$module_id)
crew_upgrade(&$messages,$module_id)
crew_uninstall(&$messages,$module_id)
crew_demodata(&$messages,$module_id,&$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wasmod_crew
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: crew_install.php,v 1.7 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function crew_demodata(&\$messages, \$module_id, &\$config, \$manifest) [line 179]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table
- *array* **&\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this module

add demonstration data to the system

this routine goes through the motions to install a (demo) workshop. However, since configuration of the required websocket server is non-trivial, the demo is not complete. It does show that the module actually exists.

Note If the module is installed via the Install Wizard, this routine is called. However, if a module is installed as an additional module after installation, the `{ $mode }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

bool function crew_install(&\$messages, \$module_id) [line 63]

Function Parameters:

- *array &\$messages* collects the (error) messages
- *int \$module_id* the key for this module in the modules table

install the module

this routine installs the module. For this module there are a few properties that need to be stored in the main `modules_properties` table. The specific table for this module is already created based on the `tabledefs`; see `install/crew_tabledefs.php`.

Note that the record for this module is already created in the `modules` table; the pkey is `$module_id`.

bool function crew_uninstall(&\$messages, \$module_id) [line 151]

Function Parameters:

- *array &\$messages* collects the (error) messages
- *int \$module_id* the key for this module in the modules table

uninstall the module

this is a hook for future extensions of Website@School. For now we simply return success. Any real code could look like this: `DELETE FROM workshops`; to delete all existing data (but who would want that). For now this is simply a nop.

Note that bluntly deleting from the `workshops` table might lead to nodes without a valid module. The better way to do it would be something like this:

```
SELECT count(node_id) AS number_of_nodes FROM workshops;
```

```
if ($number_of_nodes > 0) then
    $messages[] = 'There are still $number_of_nodes nodes with a workshop';
    return FALSE;
```

which in fact means that the table should already be empty before we can empty it. Oh well...

bool function crew_upgrade(&\$messages, \$module_id) [line 121]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

upgrade the module

this routine performs an upgrade to the installed module. Note that this module does not need any upgrade at all because there never was an earlier version.

However, if there was to be a newer version of this module, this routine is THE place to bring the database up to date compared with the existing version. For example, if an additional field 'crew_extension' was to be added to the crew-table, it could be added using a suitable (default) value, e.g. an empty string or whatever

Any existing crew records could then be updated here to fill the new field with data, e.g.

UPDATE workshops SET crew_extension = "";

etcetera. For now this routine is a nop.

crew_manifest.php

/program/modules/crew/crew_manifest.php - description of the crew module

This file defines the various components of the crew module such as the names of the various scripts and version information. This file is used when this module is installed.

- **Package** wasmod_crew
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: crew_manifest.php,v 1.11 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

crew_search.php

/program/modules/crew/crew_search.php - interface to the search-part of this module

This file defines the interface with this module for searching content. The interface consists of a (temporary) table and this function:

```
bool crew_search(&$hits,&$results,$qwords, $limit, $offset)
```

The function is called whenever module content is to be searched. At that point the helper table, which is called 'search_nodes', exists and is populated.

- **Package** wasmod_crew
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: crew_search.php,v 1.5 2016/04/05 14:33:01 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool/array function crew_search(&\$hits, &\$results, \$qwords, \$limit, [\$offset = 0]) [line 51]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results

search this module's content in selected nodes for keywords in \$qwords

This interface is documented in [htmlpage_search\(\)](#) in [htmlpage_search.php](#).

- **TODO** should we take status=0 into account too (individual entry to the workshop)? it looks like a lot of effort for a low yield

crew_view.php

/program/modules/crew/crew_view.php - interface to the view-part of the crew module

This file defines the interface with the crew-module for viewing content. The interface consists of this function:

```
crew_view(&$output,$area_id,$node_id,$module)
```

This function is called from /index.php when the node to display is connected to this module.

- **Package** wasmod_crew
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: crew_view.php,v 1.7 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

CREW_MAX_DOCUMENT_SIZE = 65536 *[line 37]*

void function crew_screen(\$loc, \$uid, \$user, \$uri, \$shop, \$org, \$date, \$hmac, \$css, \$jscript, \$progrex) *[line 467]*

Function Parameters:

- **\$loc**
- **\$uid**
- **\$user**
- **\$uri**
- **\$shop**
- **\$org**
- **\$date**
- **\$hmac**
- **\$css**
- **\$jscript**

- **\$progcrew**

construct triple-indirect edit screen in pop-up

this routine constructs the necessary HTML-code to show in the CREW editor screen. We use this dynamic construction in order to be able to simply plugin parameters and translated strings into the generated pop-up screen. Perhaps there is a better way (without using a separate javascript translation file for every language) but I haven't thought of it yet.

So here's the deal. We use PHP to generate HTML code which includes the javascript configuration for the CREW editor. This generated HTML is written to the regular page that is generated whenever the user presses the Edit-button in the regular page in the form of a long Javascript string which is used to pop-up a new window. Overly complicated, so let me know if there is a better (cleaner) way.

bool function crew_view(&\$theme, \$area_id, \$node_id, \$module) [line 54]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

display the content of the workshop linked to node \$node_id

- **TODO** FixMe: we need to take parent node permissions into account as soon as we can assign crew permissions to sections. We now specifically look at permissions in table `acls_modules_nodes` OR at global (guru) permissions for modules in table `acls`. (June 2013).
- **TODO** FixMe: we should rename SaveEdit and Save to Save and Done (May 2014)
- **TODO** FixMe: we should somehow add a csrftoken into the mix (May 2014)

array function crew_view_dialogdef() [line 207]

construct an option to select a skin and start an Edit session

this defines a dialog where the user can pick a skin from a list of existing skins and subsequently press [Edit] to actually edit the document.

bool|array function `crew_view_get_workshop_data($node_id)` [line 603]

Function Parameters:

- *int* **\$node_id** which one are we looking at?

retrieve the current version of the document + other workshop data

bool function `crew_view_save_document(&$theme, $node_id)` [line 250]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$node_id** key to the page/workshop we are working in

save the POST'ed version of the document to the workshop database

this saves the document after validating it. The maximum length is arbitrary but it should be enough for a 'normal' document.

bool function `crew_view_show_edit(&$theme, $module_id, [$first = FALSE])` [line 352]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$module_id** identifies the crew module (need that for getting module properties)
- *bool* **\$first** if TRUE we generate code to generate a popup

show the (visually almost) empty page and load or continue with the JS popup window

this routine is responsible for showing an 'empty' page and maybe for generating a JS popup window (if `$first==TRUE`). The 'empty' page contains only a form with single textarea. However, this textarea is not displayed (`display:none`) so the casual user sees nothing (but obviously without CSS it is a different matter). This textarea is used by the CREW code to store the edited document before submitting the form. Since there are no buttons of any kind, it is completely up to the JS code to generate the necessary DOM elements that are required to successfully save the document.

If `$first` is TRUE, we have to setup the popup window. This is quite complicated because

generate the necessary JS-code at runtime using JS. One of the reasons is that I want to set the correct translations in the popup window. There may be an easier way.

The Websocket protocol is used to talk to the Websocket server which is configured for this site. This setting can be manipulated using the Module Manager. In order to authenticate ourselves against the websocket server we use the following mechanism. There are a few important variables used in authenticating:

- `$origin`: this is the website's hostname as seen by the user's browser
- `$request_uri`: a string that uniquely identifies the node within the origin
- `$full_name`: the full name of the current user (ie. `$USER->full_name`)
- `$username`: the (short) name/userid of the current user (ie. `$USER->username`)
- `$request_date`: the current time (GMT) in the format "yyyy-mm-dd hh:mm:ss".
and also
- `$secret_key`: a secret shared with the Websocket server
- `$location`: the URL of the Websocket server

The authentication works as follows. The variables `$origin`, `$request_uri`, `$full_name`, `$username` and `$request_date` are concatenated in a `$message`. Then the `$message` and the `$secret_key` are used to calculate a hashed message authentication code (HMAC) according to RFC2104 (see function `{@see hmac()}` in `waslib.php`).

When connecting to the Websocket server the parameters `$request_uri`, `$full_name`, `$username` and `$request_date` are sent, together with the HMAC. The server then calculates the HMAC too and if it matches the HMAC that was sent, access is granted.

Note that the variable `$origin` is only used here to calculate the HMAC; it is not sent to the Websocket server like the other parameters. Instead we use the Origin as seen by the user's web browser. Obviously the two should match or else authentication fails. This way we check the browser's idea of where the web page is located. Also note that we made the current date/time part of the HMAC. That is done to prevent replay-attacks (the other variables are quasi-static between CREW editing sessions). It is up to the Websocket server to determine if the timestamp is (still) valid or not. This depends on a certain clock synchronisation between the webserver and the Websocket server.

Also note that the shared secret never leaves the webserver, only the hashed message is sent from webserver to Websocket server. However, the secret has to be the same on both ends.

bool function `crew_view_show_view(&$theme, $writer, $node_id)` [line 149]

Function Parameters:

- *object* **`&$theme`** collects the (html) output
- *bool* **`$writer`** if TRUE we add an Edit button to the display
- *int* **`$node_id`** key to the page we're generating

display the current version of the document and maybe an Edit button

this fetches a fresh version of the document from the database and displays it, with the header and the introduction (if any).

If the writer flag is TRUE, we also display the date/time/user of last update to the document plus we generate an Edit button which allows this user to actually edit the document via CREW.

Note Since CREW requires JavaScript, we use a trick: we initially hide the Edit button (and the Skin listbox) by setting the display-parameter of the surrounding DIV to none. Subsequently we set that parameter to 'block' using JavaScript. That means if JavaScript is not enabled, the user will not see the DIV and hence the button. If JS is enabled there is no problem. As an added bonus we also have a single line of text warning the user about JavaScript via a NOSCTIPT-tag. The only thing that can happen now is that the browser does not support the Websocket protocol. This is dealt with in the actual CREW code.

crew_tabledefs.php

/program/modules/crew/install/crew_tabledefs.php - data definition for module

- **Package** wasmod_crew
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: crew_tabledefs.php,v 1.4 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/en/crew.php - translated messages for module (English)

- **Package** wasmod_crew
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: crew.php,v 1.10 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

crew.php

/program/modules/crew/languages/nl/crew.php - translated messages for module (Dutch)

- **Package** wasmod_crew
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: crew.php,v 1.9 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wasmod_guestbook Procedural Elements

guestbook_admin.php

/program/modules/guestbook/guestbook_admin.php - management interface for module

This file defines the administrative interface to this module. The interface consists of the following four functions.

```
guestbook_disconnect(&$output,$area_id,$node_id,$module)
guestbook_connect(&$output,$area_id,$node_id,$module)
guestbook_show_edit(&$output,$area_id,$node_id,$module,$viewonly,$edit_again,$href)
guestbook_save(&$output,$area_id,$node_id,$module,$viewonly,&$edit_again)
```

These functions are called from pagemanagerlib.php whenever necessary.

- **Package** wasmod_guestbook
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: guestbook_admin.php,v 1.6 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function guestbook_connect(&\$output, \$area_id, \$node_id, \$module) [*line 71*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which we need to connect

- *array* **\$module** the module record straight from the database

connect this module to a node

this makes the link between the node \$node_id in area \$area_id and this module. It depends on the module what this function should do. Often it simply boils down to a no-op returning TRUE to indicate success.

bool function guestbook_disconnect(&\$output, \$area_id, \$node_id, \$module) [*line 52*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node from which we need to disconnect
- *array* **\$module** the module record straight from the database

disconnect this module from a node

this breaks the link between the node \$node_id in area \$area_id and this module. It depends on the module what this function should do. Often it simply boils down to a no-op returning TRUE to indicate success.

bool function guestbook_save(&\$output, \$area_id, \$node_id, \$module, \$viewonly, &\$edit_again, \$edit_again) [*line 153*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which we need to connect
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- **&\$edit_again**

save the modified content data of this module linked to node \$node_id

this validates and saves the data that was submitted by the user. If validation fails, or storing the data doesn't work, the flag \$edit_again is set to TRUE and the return value is FALSE.

If the user has cancelled the operation, the flag \$edit_again is set to FALSE and the return value is also FALSE.

If the modified data is stored successfully, the return value is TRUE (and the value of \$edit_again is a don't care).

Here is a summary of return values.

- `retval = TRUE` ==> data saved successfully
- `retval = FALSE && edit_again = TRUE` ==> re-edit the data, show the edit dialog again
- `retval = FALSE && edit_again = FALSE` ==> cancelled, do nothing

bool function guestbook_show_edit(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href) [*line 104*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

present the user with a dialog to modify the content that is connected to node \$node_id

this prepares a dialog for the user filled with existing data (if any), possibly allowing the user to modify the content. If the flag \$viewonly is TRUE, this routine should only display the content rather than let the user edit it. If the flag \$edit_again is TRUE, the routine should use the data available in the \$_POST array, otherwise it should read the data from the database (or wherever the data comes from). The parameter \$href is the place where the form should be POST'ed.

The dialog should be added to the \$output object. Useful routines are:

```
$output->add_content($content): add $content to the content area  
$output->add_message($message): add $message to the message area (feedback to the user)  
$output->add_popup_bottom($message): make $message popup in the browser after loading the page (uses  
javascript)  
$output->add_popup_top($message): make $message popup in the browser before loading the page (uses  
javascript)
```

guestbook.php

/program/modules/guestbook/languages/en/guestbook.php - translated messages for module (English)

- **Package** wasmod_guestbook
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: guestbook.php,v 1.6 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

guestbook.php

/program/modules/guestbook/languages/nl/guestbook.php - translated messages for module (Dutch)

- **Package** wasmod_guestbook
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: guestbook.php,v 1.6 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wasmod_htmlpage Procedural Elements

htmlpage_admin.php

/program/modules/htmlpage/htmlpage_admin.php - management interface for **htmlpage-module**

This file defines the administrative interface to this module. The interface consists of the following four functions.

```
htmlpage_disconnect(&$output,$area_id,$node_id,$module)
htmlpage_connect(&$output,$area_id,$node_id,$module)
htmlpage_show_edit(&$output,$area_id,$node_id,$module,$viewonly,$edit_again,$href,$option)
htmlpage_save(&$output,$area_id,$node_id,$module,$viewonly,&$edit_again,$option)
```

These functions are called from pagemanagerlib.php whenever necessary.

- **Package** wasmod_htmlpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: htmlpage_admin.php,v 1.8 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function htmlpage_connect(&\$output, \$area_id, \$node_id, \$module) [*line 73*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which we need to connect

- *array* **\$module** the module record straight from the database

connect this module to a node

this makes the link between the node \$node_id in area \$area_id and this module. In this case we simply link a data container to node \$node_id in a 1-to-1 relation. Note that we might decide later on to keep versions of pages around, e.g. by inserting new records every save rather than updating the existing.

bool function `htmlpage_disconnect(&$output, $area_id, $node_id, $module)` [line 53]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node from which we need to disconnect
- *array* **\$module** the module record straight from the database

disconnect this module from a node

this breaks the link between the node \$node_id in area \$area_id and this module. For now we simply delete the record with page data. In a future version we might want to retain the page data, 'for future reference' (but: what do we do with it? Oh well).

array function `htmlpage_get_dialogdef($viewonly)` [line 248]

Function Parameters:

- *bool* **\$viewonly** if TRUE make dialog readonly (and suppress [Save] / [Done])

construct the dialog definition for editing a plain HTML page

- **TODO** should we add a title here?

bool function `htmlpage_save(&$output, $area_id, $node_id, $module, $viewonly, &$edit_again, $option)` [line 186]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

save the modified content data of this module linked to node \$node_id

this validates and saves the data that was submitted by the user. If validation fails, or storing the data doesn't work, the flag \$edit_again is set to TRUE and the return value is FALSE.

If the user has cancelled the operation, the flag \$edit_again is set to FALSE and the return value is also FALSE.

If the modified data is stored successfully, the return value is TRUE. The value of \$edit_again is determined by the key used to save: [Save] brings the user back to edit (\$edit_again = TRUE) where [Done] leaves editing altogether (\$edit_again = FALSE).

Here is a summary of return values.

\$retval | \$edit_again | Action

----- | ----- | -----

FALSE	FALSE	[Cancel] Stop editing, unlock & return to tree view
FALSE	TRUE	[Save] or [Done] failed: redo the dialog and let user correct mistakes
TRUE	FALSE	[Done] pressed, everything saved succesfully, we're done
TRUE	TRUE	[Save] pressed, everything saved succesfully but continue editing

The parameter \$option is not used in this module.

bool function `htmlpage_show_edit(&$output, $area_id, $node_id, $module, $viewonly, $edit_again, $href, $option)`
[line 129]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides

- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

present the user with a dialog to modify the content that is connected to node \$node_id

this prepares a dialog for the user filled with existing data (if any), possibly allowing the user to modify the content. If the flag \$viewonly is TRUE, this routine should only display the content rather than let the user edit it. If the flag \$edit_again is TRUE, the routine should use the data available in the \$_POST array, otherwise it should read the data from the database (or wherever the data comes from). The parameter \$href is the place where the form should be POST'ed.

The dialog should be added to the \$output object. Useful routines are:

```
$output->add_content($content): add $content to the content area
$output->add_message($message): add $message to the message area (feedback to the user)
$output->add_popup_bottom($message): popup $message in browser after loading the page (uses javascript)
$output->add_popup_top($message): popup $message in browser before loading the page (uses javascript)
```

Note: the field version is basically unused, although the version number gets to be incremented on save. We keep the original functionality, including picking the latest version of the page_data (via \$order = version DESC).

The parameter \$option is not used in this module.

- **TODO** get rid of 'version' field and related cruft (or do something useful with it). 2014-05-05/PF

htmlpage_cron.php

/program/modules/htmlpage/htmlpage_cron.php - interface to the cron-part of the **htmlpage module**

This file defines the interface with the htmlpage-module for cron. The interface consists of this function:

```
htmlpage_cron()
```

This function is called whenever cron determines that it is time to perform this function.

- **Package** wasmod_htmlpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: htmlpage_cron.php,v 1.7 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **TODO** change this stub into a real cron function.
- **License** [GNU AGPLv3+Additional Terms](#)

bool function htmlpage_cron() [*line 42*]

routine that is called periodically by cron

htmlpage_install.php

/program/modules/htmlpage/htmlpage_install.php - installer of the htmlpage module

This file contains the htmlpage module installer. The interface consists of these functions:

```
htmlpage_install(&$messages,$module_id)
htmlpage_upgrade(&$messages,$module_id)
htmlpage_uninstall(&$messages,$module_id)
htmlpage_demodata(&$messages,$module_id,&$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wasmod_htmlpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: htmlpage_install.php,v 1.9 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function htmlpage_demodata(&\$messages, \$module_id, &\$config, \$manifest) [*line 138*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table
- *array* **&\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this module

add demonstration data to the system

this routine is a no-op because all htmlpage demodata is already created in the main demodata-routine in `/program/install/demodata.php`. This routine is retained here as an example alias placeholder.

This routine is retained here as an example and also because a routine by this name should exist (even if it does nothing).

Note If the module is installed via the Install Wizard, this routine is called. However, if a module is installed as an additional module after installation, the `{ $mode }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

bool function `htmlpage_install(&$messages, $module_id)` *[line 63]*

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

install the module

this routine installs the module. For this module there is nothing to install, so we simply return success. The appropriate table is already created based on the tabledefs); see `install/htmlpage_tabledefs.php`.

Note that the record for this module is already created in the modules table; the pkey is `$module_id`.

bool function `htmlpage_uninstall(&$messages, $module_id)` *[line 108]*

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

uninstall the module

this is a hook for future extensions of Website@School. For now we simply return success. Any real code could look like this: `DELETE FROM htmlpage;` to delete all existing data (but who would want that). For now this is simply a nop.

bool function `htmlpage_upgrade(&$messages, $module_id)` *[line 91]*

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

upgrade the module

this routine performs an upgrade to the installed module. Note that this minimalistic 'htmlpage' module does not need any upgrade at all because there never was an earlier version.

However, if there was to be a newer version of this module, this routine is THE place to bring the database up to date compared with the existing version. For example, if an additional field 'page_data_backup' was to be added to the htmlpage-table, it could be added using a suitable (default) value, e.g. the current contents of 'page_data'.

Any existing htmlpage could then be updated here to fill the new field with data, e.g.

```
UPDATE htmlpage SET page_data_backup = page_data;
```

etcetera. For now this routine is a nop.

htmlpage_manifest.php

/program/modules/htmlpage/htmlpage_manifest.php - description of the htmlpage module

This file defines the various components of the htmlpage-module such as the names of the various scripts and version information. This file is used when this module is installed.

- **Package** wasmod_htmlpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: htmlpage_manifest.php,v 1.15 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage_search.php

/program/modules/htmlpage/htmlpage_search.php - interface to the search-part of this module

This file defines the interface with this module for searching content. The interface consists of a (temporary) table and this function:

```
bool htmlpage_search(&$hits,&$results,$qwords, $limit, $offset)
```

The function is called whenever module content is to be searched. At that point the helper table, which is called 'search_nodes', exists and is populated.

- **Package** wasmod_htmlpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: htmlpage_search.php,v 1.8 2016/04/04 06:59:17 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool|array function htmlpage_search(&\$hits, &\$results, \$qwords, \$limit, [\$offset = 0]) [line 135]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results

search all htmlpages linked to selected nodes for keywords in \$qwords **TASKS**

This function needs to perform these tasks.

- If (\$results < 0) on entry, the function must calculate and set

\$results to the number of results in this module's data (could be 0).

- If (\$limit > 0) on entry, the function must return at most \$limit search hits in the \$hits array.
- If (\$results > 0) on entry the function is allowed to use that number as a hint of the total # of hits in this module's data.
- If (\$results == 0) on entry the function can decide to return immediately because (according to the caller) there are no hits anyway.

KEYWORDS TO SEARCH FOR

On entry the array \$qwords contains the sanitised user-supplied keywords, as follows:

```
$qword[$i] = array($original, $utf8lower, $quoted);  
  
$original: exactly what the user entered into the search box.  
$utf8lower: lowercase version of $original  
$quoted: escaped/quoted $original ready for LIKE '%$quoted%'
```

HELPER TABLE

On entry the table 'search_nodes' [*] exists and is populated with node_id's that are qualified to be searched. The structure of this table is as follows:

```
CREATE TABLE $search_nodes (node_id int(11) NOT NULL DEFAULT '0' PRIMARY KEY (node_id));
```

[*] Note that as with every other table the name of the table is prefixed with the prefix found in \$DB->prefix or \$CFG->prefix.

The search results of this module can be limited to the qualifying nodes by using an inner join, e.g.

```
SELECT m.data FROM $search_nodes s INNER JOIN $module m USING (node_id) WHERE $where;
```

RETURNING SEARCH RESULTS

If there are hits, these are collected in \$hits as follows:

```
$hits[] = array('title' => $title,  
               'context' => array($snippet1, $snippet2, ...),  
               'url' => $url);
```

Eventually this yields HTML-code like this:

```
<div>  
  <div><a href="$url">$title</a></div>  
  <div>$snippet1 $snippet2 ...</div>
```

```
<div><a href="$url">$url</a></div>
</div>
```

Tip: function [search_context\(\)](#) can be used to extract context snippets from hits.

If search is not relevant to the module, the NOP can be this:

```
$results = 0;
return TRUE;
```

See also [search_nodes\(\)](#).

- **Uses** [search_where\(\)](#)
- **Uses** [search_context\(\)](#)

htmlpage_view.php

/program/modules/htmlpage/htmlpage_view.php - interface to the view-part of the **htmlpage module**

This file defines the interface with the htmlpage-module for viewing content. The interface consists of this function:

```
htmlpage_view(&$output,$area_id,$node_id,$module)
```

This function is called from /index.php when the node to display is connected to this module.

- **Package** wasmod_htmlpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: htmlpage_view.php,v 1.7 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function htmlpage_view(&\$theme, \$area_id, \$node_id, \$module) [*line 46*]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

display the content of the htmlpage linked to node \$node_id

htmlpage_tabledefs.php

/program/modules/htmlpage/install/htmlpage_tabledefs.php - data definition for module

- **Package** wasmod_htmlpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: htmlpage_tabledefs.php,v 1.7 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/en/htmlpage.php - translated messages for module (English)

- **Package** wasmod_htmlpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: htmlpage.php,v 1.9 2016/03/23 09:28:22 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

htmlpage.php

/program/modules/htmlpage/languages/nl/htmlpage.php - translated messages for module (Dutch)

- **Package** wasmod_htmlpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: htmlpage.php,v 1.7 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wasmod_mailpage Procedural Elements

mailpage_tabledefs.php

/program/modules/mailpage/install/mailpage_tabledefs.php - data definition for module

Two tables are defined: mailpages and mailpages_addresses. The former contains the configuration data for a mailpage (header, introduction, etc.) and the latter is used to store 1 or more destination addresses. The records in both tables are linked to the page via node_id. The sort order in the addresses is determined by the sort_order field.

- **Package** wasmod_mailpage
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: mailpage_tabledefs.php,v 1.3 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage.php

/program/modules/mailpage/languages/en/mailpage.php - translated messages for module (English)

- **Package** wasmod_mailpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: mailpage.php,v 1.16 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage.php

/program/modules/mailpage/languages/nl/mailpage.php - translated messages for module (Dutch)

- **Package** wasmod_mailpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: mailpage.php,v 1.11 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage_admin.php

/program/modules/mailpage/mailpage_admin.php - management interface for mailpage-module

This file defines the administrative interface to this module. The interface consists of the following four functions.

```
mailpage_disconnect(&$output,$area_id,$node_id,$module)
mailpage_connect(&$output,$area_id,$node_id,$module)
mailpage_show_edit(&$output,$area_id,$node_id,$module,$viewonly,$edit_again,$href,$option)
mailpage_save(&$output,$area_id,$node_id,$module,$viewonly,&$edit_again,$option)
```

These functions are called from pagemanagerlib.php whenever necessary.

- **Package** wasmod_mailpage
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: mailpage_admin.php,v 1.11 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function mailpage_connect(&\$output, \$area_id, \$node_id, \$module) [*line 78*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which we need to connect
- *array* **\$module** the module record straight from the database

connect this module to a node

this makes the link between the node \$node_id in area \$area_id and this module. In this case we create a single 'mailpages' record linked to node_id. Any addresses can be added later as desired and records in 'mailpages_addresses' will be created as necessary.

bool function mailpage_dialog_validate_address(&\$dialogdef) [*line 695*]

Function Parameters:

- *object* **&\$dialogdef** defines the dialog and will hold POSTed values

validate the data entered by the user

Other than the standard validation we check for at least 1 @-sign in the email field As a side effect, the dialogdef is filled with the POST'ed values.

bool function mailpage_disconnect(&\$output, \$area_id, \$node_id, \$module) [*line 55*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node from which we need to disconnect
- *array* **\$module** the module record straight from the database

disconnect this module from a node

this breaks the link between the node \$node_id in area \$area_id and this module. For now we simply delete the records in 'mailpages' and 'mailpages_addresses' linked to node node_id. Note that the order is not important (in this case) because there are no FK constraints between the two tables. However, there are FK constraints with the nodes table. The db_delete() here is a simple best effort.

array function mailpage_get_addresses(\$node_id) [*line 943*]

Function Parameters:

- *int* **\$node_id** indicates page

retrieve current list of addresses in an array (could be empty)

this retrieves the addresses associated with \$node_id from the database. As an important side effect all entries' sort order values are renumbered. This means that we always have a neat set of sort order numbers 10, 20, ...

On error we return an empty array as a sort of best effort. The error is logged though.

array function mailpage_get_dialogdef_address(&\$output, \$viewonly, \$node_id, \$address_id, [\$sort_order = 10])
[line 806]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$viewonly** if TRUE the Save button is not displayed and config values cannot be changed
- *int* **\$node_id** identifies the current mailpage
- *int* **\$address_id** identifies the address, 0 for a new one
- *int* **\$sort_order** is the next available sort order for new records

construct a dialog definition for a mailpage destination address (add/edit)

this constructs a dialog definition which handles adding and editing destination addresses. In the case of adding a new address we start with empty fields (mostly), otherwise we attempt to load the existing data from the DB.

array function mailpage_get_dialogdef_config(&\$output, \$viewonly, \$node_id) [line 721]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$viewonly** if TRUE the Save button is not displayed and config values cannot be changed
- *int* **\$node_id** identifies the current mailpage

construct a dialog definition for the main mailpage configuration

this constructs the dialogdef and also fills it with data from the DB (if any)

array function mailpage_get_dialogdef_delete(\$viewonly) [line 918]

Function Parameters:

- *int* **\$viewonly** if TRUE the Save button is not displayed and config values cannot be changed

construct a dialog definition for deletion of a destination address

this constructs a dialog definition which handles deleting of a destination address. There

is not much to do: the address_id to delete is conveyed via \$_GET[].

array function mailpage_get_icon_delete(&\$output, \$node_id, \$viewonly, \$address_id, \$aparams, \$aparms) [line 987]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** identifies the current mailpage
- *int* **\$viewonly** if TRUE the Save button is not displayed and config values cannot be changed
- *int* **\$address_id** identifies the address, 0 for a new one
- *int* **\$aparms** contains values to substitute in translations, e.g. {ADDRESS_ID}
- **\$aparams**

construct a delete button for deletion of a destination address

this constructs a clickable icon/button that leads to the delete confirmation dialog. We use the same code as does the Page Manager (picking up the icon from the current skin et al). This makes that the icon automagically adapts to the skin, eg. the textonly skin.

bool function mailpage_save(&\$output, \$area_id, \$node_id, \$module, \$viewonly, &\$edit_again, \$option) [line 202]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$option** the selected submenu option or NULL if user used the 'Content' option to get here

dispatcher for save routines

We use different subroutines for clarity.

bool function mailpage_save_configuration(&\$output, \$area_id, \$node_id, \$module, \$viewonly, &\$edit_again) [line 259]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise

save the modified configuration data of this mailpage linked to node \$node_id

bool function mailpage_save_destinations(&\$output, \$area_id, \$node_id, \$module, \$viewonly, &\$edit_again, \$edit_again) [line 570]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- **&\$edit_again**

process the POSTed destination address data

this is another complex routine to fit saving of three different dialogs into one. See also the comprehensive documentation in function [mailpage_show_edit_destinations\(\)](#).

Basically this routine handles all database operations like inserting new addresses, editing or deleting existing ones. We arrive here because the user has either pressed [Save] or [Done], indicating that the data should be stored, or [Cancel], indicating the wish to end the adding/editing/deleting operation (dialogs #2 and #3 as we called them in function

[mailpage_show_edit_destinations\(\)](#)).

However, there is one other [Cancel] button: the one in the destinations addresses overview (dialog #1). This button ends the editing of the node altogether and returns to the Page Manager. This specific situation is indicated by returning FALSE after setting \$edit_again to FALSE. All other cases always return FALSE with \$edit_again set to TRUE.

This means that we do have to check for the validity of the dialog data in show_edit() again, because we abuse the returned value to stay in the overview screen (dialog #1). This is ugly, and I am sorry about that.

Here too we distinguish between add/edit and delete via address_id and delete_id respectively (linked to \$_GET['address'] and \$_GET['delete']). We do check for invalid address_id's: if we don't recognise the address_id/delete_id we simply do nothing (but send a message to the log nevertheless).

bool function mailpage_show_content(&\$output, \$node_id, \$module) [line 154]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

show the main content screen (in fact documentation only)

this displays the simple overview screen with documentation. It ends with a cancel button that ends the whole edit operation for this node.

bool function mailpage_show_edit(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href, \$option) [line 124]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database

- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *string* **\$option** the selected submenu option or NULL if user used the 'Content' option to get here

dispatcher for presenting the user with one of the mailpage dialogs

this routine looks at \$option to see which dialog is requested and subsequently displays that dialog.

There are three possibilities:

- \$option == NULL: an overview/help screen basically pointing the user to the submenu items -
 \$option == 'configuration': main configuration via table 'mailpages' - \$option == 'destinations':
 destination addresses config via 'mailpages_addresses'

We use different subroutines for clarity.

bool function mailpage_show_edit_configuration(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href) [line 232]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

present the user with the main configuration dialog for mailpage at node \$node_id

bool function mailpage_show_edit_destinations(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href) [line 388]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides

- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

present the user with a dialog for mailpage destination addresses at node \$node_id

this is a complex routine that is responsible for constructing and showing the dialogs associated with destination addresses.

The following dialogs exist. #1. Mailpage Destinations Overview

Add a new destination address

[D] Principal (10)

[D] Webmaster (20)

[Cancel]

#2. Add new destination address/Edit existing destination address

Name: _____

E-Mail: _____

Description: _____

Thankyou: _____

Sort order: _____

[Save] [Done] [Cancel]

#3. Delete address confirmation

[Delete] [Cancel]

There are some complex interactions between these screens and the corresponding save routine [mailpage_save_destinations\(\)](#). The effects we try to achieve are as follows. The link 'Add a new destination address' leads to dialog #2 with address_id = 0 The links 'Principal' and 'Webmaster' lead also to dialog #2 with valid, existing address_id != 0 The delete button '[D]' leads to dialog #3 with valid, existing delete_id != 0

The [Cancel] button in dialog #1 ends the editing of this node; ie. returns to Page Manager. The other [Cancel] buttons bring the user back to dialog #1. In effect this is a kind of zoom function: you can zoom in on an address and go back to the list without leaving this node, etc. etc.

The [Done] and [Delete] buttons perform an action (saving or deleting a destination) and also return to dialog #1.

The [Save] button always has the meaning "save the data and continue editing". For the Edit dialog #2 this works perfect. For the Add dialog #2 there is a slight problem, because the Add dialog received `address_id = 0` and saving the new address changes that: the `address_id` should now be the `last_id` as inserted in the database instead of 0. Unfortunately there is no clean way to convey this fact via the module interface with Page Manager. In order to provide a smooth user experience I had to use an ugly hack in the form of a global variable called `$mailpage_new_address_id`. This variable is usually set to `NULL`, but after successful insertion into the DB of the new address, the new `address_id` is stored here (from the `save()` routine). We pick it up here and use that value instead of the value 0 and present the user with an Edit dialog #2 for the newly added address when she presses the [Save] button. This magic works because we know that after Page Manager calls the `save()` routine, which sets the `$edit_again` flag, Page Manager will call `show_view()` again. Ugly, but it works. Sorry.

Note that we also take `$viewonly` into account: if this flag is set, there is no 'Add a destination' link because `$viewonly` does not allow anything to be changed in the node configuration, including adding (and editing and deleting) destinations.

Note: the parameter that holds the `address_id` is called `$_GET['address']` in case of adding or editing an address. If we are deleting we use `$_GET['delete']` in order to distinguish these cases. I use the variable name `$delete_id` to keep things separated.

bool function mailpage_show_edit_destinations_overview(&\$output, \$node_id, \$viewonly, \$href, \$records) [line 490]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *array* **\$records** the currently defined destination addresses

show a screen with a links to add/edit/delete destination addresses

mailpage_cron.php

/program/modules/mailpage/mailpage_cron.php - interface to the cron-part of the mailpage module

This file defines the interface with the mailpage-module for cron. The interface consists of this function:

```
mailpage_cron()
```

This function is called whenever cron determines that it is time to perform this function.

- **Package** wasmod_mailpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: mailpage_cron.php,v 1.4 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function mailpage_cron() [*line 44*]

routine that is called periodically by cron

there is nothing in this module that needs to be done by cron, so this routine is basically a nop.

mailpage_install.php

/program/modules/mailpage/mailpage_install.php - installer of the mailpage module

This file contains the mailpage module installer. The interface consists of these functions:

```
mailpage_install(&$messages,$module_id)
mailpage_upgrade(&$messages,$module_id)
mailpage_uninstall(&$messages,$module_id)
mailpage_demodata(&$messages,$module_id,&$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wasmod_mailpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: mailpage_install.php,v 1.6 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function mailpage_demodata(&\$messages, \$module_id, &\$config, \$manifest) [*line 139*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table
- *array* **&\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this module

add demonstration data to the system

this routine is a no-op because there is no mailpage demodata and if it is there, it is already created in the main demodata-routine in /program/install/demodata.php.

This routine is retained here as an example and also because a routine by this name should exist (even if it does nothing).

Note If the module is installed via the Install Wizard, this routine is called. However, if a module is installed as an additional module after installation, the `{ $mode }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

bool function mailpage_install(&\$messages, \$module_id) [line 61]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

install the module

this routine installs the module. For this module there is nothing to install, so we simply return success.

Note that the record for this module is already created in the modules table; the pkey is `$module_id`.

bool function mailpage_uninstall(&\$messages, \$module_id) [line 109]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

uninstall the module

this is a hook for future extensions of Website@School. For now we simply return success.

bool function mailpage_upgrade(&\$messages, \$module_id) [line 77]

Function Parameters:

- *array* **&\$messages** collects the (error) messages

- *int* **\$module_id** the key for this module in the modules table

upgrade the module

this routine performs an upgrade to the installed module. This updates the list of submenu options in the module record. The submenu-feature was added 2014-05-05 and appears in release 0.90.6.

mailpage_manifest.php

/program/modules/mailpage/mailpage_manifest.php - description of the mailpage module

This file defines the various components of the mailpage module such as the names of the various scripts and version information. This file is used when this module is installed.

- **Package** wasmod_mailpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: mailpage_manifest.php,v 1.10 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

mailpage_search.php

/program/modules/mailpage/mailpage_search.php - interface to the search-part of this module

This file defines the interface with this module for searching content. The interface consists of a (temporary) table and this function:

```
bool mailpage_search(&$hits,&$results,$qwords, $limit, $offset)
```

The function is called whenever module content is to be searched. At that point the helper table, which is called 'search_nodes', exists and is populated.

- **Package** wasmod_mailpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: mailpage_search.php,v 1.5 2016/04/05 14:33:01 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool|array function mailpage_search(&\$hits, &\$results, \$qwords, \$limit, [\$offset = 0]) [line 49]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results

search this module's content in selected nodes for keywords in \$qwords

This interface is documented in [htmlpage_search\(\)](#) in [htmlpage_search.php](#).

mailpage_view.php

/program/modules/mailpage/mailpage_view.php - interface to the view-part of the mailpage module

This file defines the interface with the mailpage-module for viewing content. The interface consists of this function:

```
mailpage_view(&$output,$area_id,$node_id,$module)
```

This function is called from /index.php when the node to display is connected to this module.

- **Package** wasmod_mailpage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: mailpage_view.php,v 1.9 2016/03/23 09:28:23 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
MAILPAGE_REFERENCE = sha1(__FILE__."":__LINE__) [line 37]
```

```
bool function mailpage_send_message($config, $dialogdef, $ip_addr, $delay) [line 488]
```

Function Parameters:

- *array* **\$dialogdef** array that defines the data fields including values
- *string* **\$ip_addr** the originating IP-address
- *int* **\$delay** the # of seconds since time=t0
- *array* **\$config** mailpage configuration data in a (nested) array

actually send the visitor's message to the selected destination

In order to get a feeling for the time a visitor needs, we also record the delay (in seconds) next to the visitor's IP address.

- **TODO** extra validation of set_mailreplyto and set_subject?
- **TODO** more available parameters in subject_line?
- **TODO** make body of mail configuratble?

void function mailpage_show_form(&\$theme, \$config, \$dialogdef) [line 348]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$dialogdef** array that defines the input fields
- *array* **\$config** mailpage configuration data in a (nested) array

display the contact form

this displays the contact form. Every destination gets a separate DIV just below the listbox, with the additional information for that destination. If JavaScript is NOT enabled, all DIVs are displayed, otherwise only the currently selected destination is displayed and the others are not. IOW: this form is still usable even without JS enabled AND it is screenreader-friendly.

If there is only a single destination, the listbox is not defined in the dialogdef and hence not rendered at all: there is no point in showing a list of options if there is nothing to choose from. This means that the necessary javascript is NOT added and also no DIVs are shown. So: a clean, uncluttered form in case of a single destination.

void function mailpage_show_preview(&\$theme, \$config, \$dialogdef, \$ip_addr) [line 431]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$dialogdef** array that defines the input fields
- *string* **\$ip_addr** the originating IP-address
- *array* **\$config** mailpage configuration data in a (nested) array

show a preview of the message to the visitor

this shows a preview of the message to visitor. Nothing is editable, it is view-only. The only option is to either press the Send-button to actually send the messate OR to press the Edit button to go back to the editable form.

Sending a message is a two-step procedure by design.

void function mailpage_show_thankyou(&\$theme, \$config, \$dialogdef, \$ip_addr) [line 531]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$dialogdef** array that defines the input fields
- *string* **\$ip_addr** the originating IP-address
- *array* **\$config** mailpage configuration data in a (nested) array

thank the visitor for the message and show a text copy too

Almost the same as {*@see* mailpage_show_preview()}.

- **TODO** should we have an OK button at all???

bool function mailpage_view(&\$theme, \$area_id, \$node_id, \$module) [line 48]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

display the content of the mailpage linked to node \$node_id

bool function mailpage_view_dialog_validate(&\$dialogdef) [line 307]

Function Parameters:

- *object* **&\$dialogdef** defines the dialog and will hold POSTed values

validate the data entered by the visitor

Other than the standard validation we check for at least 1 @-sign in the email address. (It is non-trivial to validate an email address, it may not even be possible to do it in a single regular expression. See <http://stackoverflow.com/questions/201323> for more information.)

bool/array function `mailpage_view_get_config($node_id)` [line 187]

Function Parameters:

- *int* **\$node_id** identifies the page

retrieve all configuration data for this mailpage

this retrieves all configuration data for this mailpage, i.e. both the general parameters (header/intro/etc.) and the full list of configured destination addresses. Here is a quick reminder of the structure in \$config. 'node_id'

'header'

'introduction'

'message'

'addresses'

['mailpage_address_id','node_id','sort_order','name','email','description','thankyou'],...

=

array function `mailpage_view_get_dialogdef($config, $token_key)` [line 220]

Function Parameters:

- *array* **\$config** mailpage configuration including addresses
- *string* **\$token_key**

construct a dialog definition for the visitor's mail form

this defines the contact form. If there is but one destination we don't even add the listbox because it makes no sense to tell the user to select an option from a listbox with a single item.

`require_once $CFG->progdir."/lib/tokenlib.php"` [line 38]

Package wasmod_mypage Procedural Elements

mypage.php

/program/modules/mypage/languages/en/mypage.php - translated messages for module (English)

- **Package** wasmod_mypage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: mypage.php,v 1.8 2016/03/23 11:33:14 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

mypage.php

/program/modules/mypage/languages/nl/mypage.php - translated messages for module (Dutch)

- **Package** wasmod_mypage
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: mypage.php,v 1.7 2016/03/23 11:33:14 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

mypage_admin.php

/program/modules/mypage/mypage_admin.php - management interface for mypage-module

This file defines the administrative interface to this module. The interface consists of the following four functions.

```
mypage_disconnect(&$output,$area_id,$node_id,$module)
mypage_connect(&$output,$area_id,$node_id,$module)
mypage_show_edit(&$output,$area_id,$node_id,$module,$viewonly,$edit_again,$href,$option)
mypage_save(&$output,$area_id,$node_id,$module,$viewonly,&$edit_again,$option)
```

These functions are called from pagemanagerlib.php whenever necessary.

- **Package** wasmod_mypage
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: mypage_admin.php,v 1.6 2016/03/22 13:08:08 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function mypage_connect(&\$output, \$area_id, \$node_id, \$module) *[line 67]*

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which we need to connect
- *array* **\$module** the module record straight from the database

connect this module to a node

this makes the link between the node \$node_id in area \$area_id and this module. There is nothing to configure so this routine is a NOP.

bool function mypage_disconnect(&\$output, \$area_id, \$node_id, \$module) *[line 51]*

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node from which we need to disconnect
- *array* **\$module** the module record straight from the database

disconnect this module from a node

this breaks the link between the node \$node_id in area \$area_id and this module. Since there is nothing to configure this is a NOP.

bool function mypage_save(&\$output, \$area_id, \$node_id, \$module, \$viewonly, &\$edit_again, \$option) [line 127]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

save the modified content data of this module linked to node \$node_id

basically also a NOP: If the user has cancelled the operation, the flag \$edit_again is set to FALSE and the return value is also FALSE.

If the modified data is stored successfully, the return value is TRUE and the value of \$edit_again is based on the button used to save ([Save] or [Done]).

Here is a summary of return values.

\$retval | \$edit_again | Action

----- | ----- | -----

FALSE | FALSE | [Cancel] Stop editing, unlock & return to tree view

FALSE | TRUE | [Save] or [Done] failed: redo the dialog and let user correct mistakes

TRUE	FALSE	[Done] pressed, everything saved succesfully, we're done
TRUE	TRUE	[Save] pressed, everything saved succesfully but continue editing

The parameter \$option is not used in this module.

bool function mypage_show_edit(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href, \$option)
[line 87]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

present the user with a dialog to modify the mypage that is connected to node \$node_id
display a short message indicating that this module has no configuration options. The user can only use the Cancel button to return to Page Manager.

mypage_cron.php

/program/modules/mypage/mypage_cron.php - interface to the cron-part of the mypage module

This file defines the interface with the mypage-module for cron. The interface consists of this function:

```
mypage_cron( )
```

This function is called whenever cron determines that it is time to perform this function.

- **Package** wasmod_mypage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: mypage_cron.php,v 1.1 2016/03/22 13:08:08 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function mypage_cron() [*line 44*]

routine that is called periodically by cron

there is nothing in this module that needs to be done by cron, so this routine is basically a nop.

mypage_install.php

/program/modules/mypage/mypage_install.php - installer of the mypage module

This file contains the mypage module installer. The interface consists of these functions:

```
mypage_install(&$messages,$module_id)
mypage_upgrade(&$messages,$module_id)
mypage_uninstall(&$messages,$module_id)
mypage_demodata(&$messages,$module_id,&$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wasmod_mypage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: mypage_install.php,v 1.1 2016/03/22 13:08:08 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function mypage_demodata(&\$messages, \$module_id, &\$config, \$manifest) [*line 133*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table
- *array* **&\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this module

add demonstration data to the system

this routine is a no-op because all mypage demodata is already created in the main demodata-routine in `/program/install/demodata.php`. This routine is retained here as an example alias placeholder.

This routine is retained here as an example and also because a routine by this name should exist (even if it does nothing).

Note If the module is installed via the Install Wizard, this routine is called. However, if a module is installed as an additional module after installation, the `{ $mode }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

bool function mypage_install(&\$messages, \$module_id) [line 61]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

install the module

this routine installs the module. For this module there is nothing to install, so we simply return success.

Note that the record for this module is already created in the modules table; the pkey is `$module_id`.

bool function mypage_uninstall(&\$messages, \$module_id) [line 103]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

uninstall the module

this is a hook for future extensions of Website@School. For now it is a NOP and we simply return success.

bool function mypage_upgrade(&\$messages, \$module_id) [line 89]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

upgrade the module

this routine performs an upgrade to the installed module. Note that this minimalistic 'mypage' module does not need any upgrade at all because there never was an earlier version.

However, if there was to be a newer version of this module, this routine is THE place to bring the database up to date compared with the existing version. For example, if an additional field 'mypage_extension' was to be added to the mypage-table, it could be added using a suitable (default) value, e.g. an empty string or whatever

Any existing mypage could then be updated here to fill the new field with data, e.g.

```
UPDATE mypage SET mypage_extension = '';
```

etcetera. For now this routine is a nop.

mypage_manifest.php

/program/modules/mypage/mypage_manifest.php - description of the mypage module

This file defines the various components of the mypage module such as the names of the various scripts and version information. This file is used when this module is installed.

- **Package** wasmod_mypage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: mypage_manifest.php,v 1.3 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

mypage_search.php

/program/modules/mypage/mypage_search.php - interface to the search-part of this module

This file defines the interface with this module for searching content. The interface consists of a (temporary) table and this function:

```
bool mypage_search(&$hits,&$results,$qwords, $limit, $offset)
```

The function is called whenever module content is to be searched. At that point the helper table, which is called 'search_nodes', exists and is populated.

- **Package** wasmod_mypage
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: mypage_search.php,v 1.2 2016/04/01 18:31:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool|array function mypage_search(&\$hits, &\$results, \$qwords, \$limit, [\$offset = 0]) [line 49]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results

search this module's content in selected nodes for keywords in \$qwords

This interface is documented in [htmlpage_search\(\)](#) in [htmlpage_search.php](#).

mypage_view.php

/program/modules/mypage/mypage_view.php - interface to the view-part of the mypage module

This file defines the interface with the mypage-module for viewing content. The interface consists of this function:

```
mypage_view(&$output,$area_id,$node_id,$module)
```

This function is called from /index.php when the node to display is connected to this module.

- **Package** wasmod_mypage
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: mypage_view.php,v 1.4 2016/06/21 06:16:58 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function mypage_authorisation(&\$dialogdef) [*line 612*]

Function Parameters:

- *object* **&\$dialogdef** defines the dialog and will hold POSTed values

check the authorisation for a submitted dialogue

check if the remote caller is not blacklisted and see if \$USER->username and \$dialogdef['password']['value'] match.

- **Uses** [login_is_blacklisted\(\)](#)
- **Usedby** [mypage_password_dialog_validate\(\)](#)
- **Uses** [login_failure_reset\(\)](#)
- **Uses** [login_failure_delay\(\)](#)

- Uses [authenticate_user\(\)](#)
- Uses [login_failure_blacklist_address\(\)](#)
- Uses [acceptable_new_password\(\)](#)

bool function mypage_password_dialog_validate(&\$dialogdef) [*line 553*]

Function Parameters:

- *object* **&\$dialogdef** defines the dialog and will hold POSTed values

validate entries in user password dialogue

- Uses [mypage_authorisation\(\)](#)
- Uses [acceptable_new_password\(\)](#)

bool/array function mypage_password_get_dialogdef() [*line 502*]

construct the edit dialog for the password of the current user

heavily leans on see [loginlib.php](#).

bool function mypage_profile_dialog_validate(&\$dialogdef, \$user_id) [*line 424*]

Function Parameters:

- *object* **&\$dialogdef** defines the dialog and will hold POSTed values
- *int* **\$user_id** user to check

validate entries in user profile dialogue

exta validation: email address should contain at least one '@' (very crude) Also the user MUST reauthorise to let the changes have effect

bool/array function mypage_profile_get_dialogdef() [*line 325*]

construct the edit dialog for the profile of the current user

user must reauthenticate to let changes take effect hence the password field

- **TODO** should we remove username and datadir altogether?
- **Uses** \$USER
- **Uses** \$LANGUAGE

bool function mypage_profile_save(&\$theme, &\$dialogdef, \$user_id) [line 462]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *object* **&\$dialogdef** defines the dialog and holds POSTed values to save
- *int* **\$user_id** user to check

save modified user profile in database and keep a copy in core

save the modified data to database and also in core (in \$USER) and even in \$_SESSION for a few selected parameters. Note that the username is always updated, even if the field is read/only in the dialogue. The source for that data is \$USER so there should be no harm.

bool function mypage_view(&\$theme, \$area_id, \$node_id, \$module) [line 63]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

display the content mypage linked to node \$node_id

there are 4 different views: one if the user is not yet logged in: 1. a login dialogue (see [mypage view login\(\)](#)) and three when the user is logged in: 2. an overview (see

[mypage_view_home\(\)](#),

3. a change profile dialogue (see [mypage_view_profile\(\)](#)), and

4. a change password dialogue (see [mypage_view_password\(\)](#)).

The idea is to present the user with a list of handy links, first links to managing the user's profile, then (if the user has privileges) links to admin.php and finally a list of areas that are accessible for the user (now she is logged in).

bool function mypage_view_home(&\$theme, \$area_id, \$node_id) [line 112]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected

show links to profile, admin.php and available areas

this personal page contains links to places the user is allowed to go.

bool function mypage_view_login(&\$theme, \$area_id, \$node_id) [line 90]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$node_id** the node to which this module is connected
- *int* **\$area_id** identifies the area where \$node_id lives

show a login dialog

build on loginlib: show login dialog. Actual processing is done in main_index() but eventually we end up at the node node_id.

bool function mypage_view_password(&\$theme, \$area_id, \$node_id) [line 269]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives

- *int* **\$node_id** the node to which this module is connected

show/process password dialogue

this dialogue requires both the old and the new password. the username is implied (because the user is logged in) The extra authentication is necessary to prevent a passer by to change a password on an unguarded screen.

The logic applied is re-used from see [loginlib.php](#), including the blacklist mechanism; it is therefore not possible to brute force a new password.

We have only 1 submit button for save: we always return to the overview screen after changing the password.

Note that the new password has to be entered twice. This excercises the muscle memory for better remembering the password.

- Uses [login_change_password\(\)](#)
- Uses [loginlib.php](#)

bool function mypage_view_profile(&\$theme, \$area_id, \$node_id) [line 202]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected

show/process profile dialogue

display a dialogue where the user can change a few properties of her own account. Notably missing: password (see see [mypage_view_password\(\)](#)). Note that currently the username and the also the directory are not editable. The information is displayed, however.

Processing is also done via this routine. The dialog ends with three submit buttons: Save, Done and Cancel. Save and Done store the new values in the user record in the database AND in the global \$USER object too. Save then shows the dialogue again, while Done returns to the overview. Cancel always returns to the overview without changing anything.

Package wasmod_newsletter Procedural Elements

newsletter_tabledefs.php

/program/modules/newsletter/install/newsletter_tabledefs.php - data definition for module

The following tables are defined:

newsletters - 1 record per newsletter, tied to a node; contains main configuration
newsletter_articles - 0, 1 or more records containing unpublished articles
newsletter_issues - 0, 1 or more records containing published newsletter issues
newsletter_subscribers - 0, 1 or more records with subscriber email addresses
newsletter_queue - 0, 1 or more records with newsletters that need to be sent

The tables are linked together via the newsletter_id which value is identical to the node_id of the page to which this instance of the module is linked.

- **Package** wasmod_newsletter
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: newsletter_tabledefs.php,v 1.6 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

newsletter.php

/program/modules/newsletter/languages/en/newsletter.php - translated messages for module (English)

- **Package** wasmod_newsletter
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: newsletter.php,v 1.36 2016/05/26 13:22:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

newsletter.php

/program/modules/newsletter/languages/en/newsletter.php - translated messages for module (Dutch)

- **Package** wasmod_newsletter
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: newsletter.php,v 1.5 2016/05/26 13:22:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

newsletter_admin.php

/program/modules/newsletter/newsletter_admin.php - management interface for newsletter-module

This file defines the administrative interface to this module. The interface consists of the following four functions.

```
newsletter_disconnect(&$output,$area_id,$node_id,$module)
newsletter_connect(&$output,$area_id,$node_id,$module)
newsletter_show_edit(&$output,$area_id,$node_id,$module,$viewonly,$edit_again,$href,$option)
newsletter_save(&$output,$area_id,$node_id,$module,$viewonly,&$edit_again,$option)
```

These functions are called from pagemanagerlib.php whenever necessary.

- **Package** wasmod_newsletter
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: newsletter_admin.php,v 1.43 2016/06/02 16:28:12 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

string function newsletter_add_related(&\$mailer, \$filename) [line 4456]

Function Parameters:

- *object* **&\$mailer** builds the email message with (related) attachments
- *string* **\$filename** identifies the local file to add

add the file in path as a related attachment to the email being built

note: mimetype of \$filename becomes application/octet-stream with disposition = attachment if we cannot determine the mimetype

string function newsletter_body2text(\$body, &\$cids) [line 4543]

Function Parameters:

- *string* **\$body** holds the body of an HTML page (newsletter)

- `array $cids` holds a list of mappings between a local path and a CID

convert the body of an HTML-document to text

this routine tries to convert the HTML in `$body` into plain text while maintaining as much as possible of the layout.

The HTML can contain links '``' and images '``'. For local files we already established CID's (see [newsletter_urls2cids\(\)](#)) and we know that those files are to be found in the body. In order to reduce the work, we first convert all those tags into the string '`[ATTACHMENT:$href]`'.

Next step is to replace the other A- and IMG-tags to their href|src.

Finally we use a heuristic approach to replace a subset of all HTML tags with a more or less readable translation. This can certainly be improved upon.

- See [newsletter_urls2cids\(\)](#)

`array function newsletter_compose_get_dialogdef($node_id, $viewonly, $article_id) [line 1558]`

Function Parameters:

- `int $node_id` identifies the current newsletter
- `int $viewonly` if TRUE the Save button is not displayed and values cannot be changed
- `int $article_id` identifies the article, 0 for a new one

construct a dialog definition for a newsletter article (add/edit)

this constructs a dialog definition which handles adding and editing articles In the case of adding a new article we start with empty fields (mostly), otherwise we attempt to load the existing data from the DB.

If the `article_id` is 0 we are adding a new article. In that case the name and the email address of the current user are automatically added. Note that the email field is write-once: we do want to record the initial value but it is not supposed to change; we want to keep a contact address for the author of the article, at least while the article exists in the articles table (the record will be deleted once an issue is officially published).

array function newsletter_compose_get_dialogdef_delete(\$viewonly) [*line 1667*]

Function Parameters:

- *int* **\$viewonly** if TRUE the Delete button is not displayed

construct a dialog definition for deletion of an article

this constructs a dialog definition which handles deleting of an article.

bool function newsletter_compose_move(\$node_id, \$article_id, \$reverse) [*line 1256*]

Function Parameters:

- *int* **\$node_id** identifies the newsletter
- *int* **\$article_id** is the article that needs to be moved
- *bool* **\$reverse** TRUE=move up in the list, FALSE is move down

swap two articles in the list of ((de)selected) articles

this swaps two adjacent articles, effectively moving article \$article_id up (or down) a position within the list. If the article is already at the end/begin of the list nothing happens.

int function newsletter_compose_new_sort_order(\$newsletter_id, [\$selected = TRUE], [\$first = FALSE]) [*line 1691*]

Function Parameters:

- *int* **\$newsletter_id** identifies the newsletter
- *bool* **\$selected** identifies which list to add to
- *bool* **\$first** if TRUE, the article is inserted at the top of the list, FALSE is append at the end

calculate a new sort order based on the current values of available articles in this newsletter

note that as a rule the new sort order is negative for unselected articles and positive for selected articles. Also, none of the existing sort_orders are touched or modified here because know that eventually the selected articles are 'eaten' once they are incorporated into a newsletter issue.

bool function newsletter_compose_save(&\$output, \$node_id, \$viewonly, &\$edit_again, \$edit_again) [line 1324]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node alias the newsletter
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- **&\$edit_again**

handle saving & deleting of articles

this handles all actual article add/edit/delete actions. criterion for delete is the existence of the delete-parameter (which identifies the article to delete), the criterion for edit/add is the article-parameter.

Part of the logic handling the [Cancel] button is done here too (see also [newsletter_compose_show_edit\(\)](#)).

bool function newsletter_compose_show(&\$output, \$node_id, \$module, \$viewonly, \$edit_again, \$href, &\$edit_again) [line 758]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$href** the action property of the HTML-form, the place where data will be POSTed
- **\$edit_again**

dispatcher for displaying newsletter composition dialogs

bool function newsletter_compose_show_add(&\$output, \$node_id, \$viewonly, \$edit_again, \$href) [line 1064]

Function Parameters:

- *object* **\$\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *bool* **\$viewonly** if TRUE, adding/editing is not allowed (but simply showing the content is)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

show the dialog for adding a new article

actual work is done in [newsletter_compose_show_edit\(\)](#) but with `article_id` set to 0 instead of a valid, existing `article_id`.

bool function newsletter_compose_show_delete(&\$output, \$node_id, \$article_id, \$viewonly, \$edit_again, \$href)
[line 1154]

Function Parameters:

- *object* **\$\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *int* **\$article_id** identifies the article to edit (0 means add a new article dialog)
- *bool* **\$viewonly** if TRUE, adding/editing is not allowed (but simply showing the content is)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

show a delete confirmation dialog

bool function newsletter_compose_show_edit(&\$output, \$node_id, \$article_id, \$viewonly, \$edit_again, \$href) [line 1101]

Function Parameters:

- *object* **\$\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *int* **\$article_id** identifies the article to edit (0 means add a new article dialog)

- *bool* **\$viewonly** if TRUE, adding/editing is not allowed (but simply showing the content is)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

show dialog for editing (or adding) an article

this displays the edit dialog for articles. It is also used by [newsletter_compose_show_add\(\)](#) with `article_id = 0`.

This dialog usually ends with three buttons: [Save] - saves the current article and edits again [Done] - save the article and return to the list of articles [Cancel] - don't save and return to the list of articles This poses a little problem when a new article is added and the user clicks [Save]; there is no elegant way to convey the new `article_id` (which was 0 in the add article dialog) to the routine that subsequently edits the newly added article. The (ugly) work around is to set the global `$newsletter_new_article_id` in the save routine and pick up the new value here.

Also part of the logic to deal with [Cancel] is handled here; the [Cancel]-button in this Add/Edit dialog (and the Delete dialog in [newsletter_compose_show_delete\(\)](#) for that matter) should return to the list of articles, whereas the [Cancel] button in the list of articles should lead to the content overview in .

bool function newsletter_compose_show_list(&\$output, \$node_id, \$viewonly, \$edit_again, \$href, &\$edit_again)
[line 841]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which the content is connected
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$href** the action property of the HTML-form, the place where data will be POSTed
- **\$edit_again**

create the main newsletter compose screen

Display a list of currently selected articles followed by a list of currently unselected articles.

`$newslettertitle` (volume \$volume number \$number)

Add an article

Currently selected articles

[D] [F] [L] Article title
[D] [F] [H] [L] Article title
...
[D] [F] [H] Article title

Preview link (in new window)
Testmail link

Currently unselected articles

[D] [C] [L] Article title
[D] [C] [H] [L] Article title
...
[D] [C] [H] Article title

[Cancel]

[D] = delete, [F] = future (postpone), [C] = current (select) [H] = higher (up), [L] lower (down)

bool function newsletter_compose_show_list_article(&\$output, \$a_params, &\$article, \$first, \$last, \$viewonly) [line 978]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *array* **\$a_params** holds information to build a href
- *array* **&\$article** properties of the article straight from the database
- *bool* **\$first** if TRUE this article is first in the list hence no 'up' icon
- *bool* **\$last** if TRUE this article is last in the list hence no 'down' icon
- *bool* **\$viewonly** if TRUE modification is not allowed so no icons at all

output various icons and links for manipulating articles

void function newsletter_compose_show_preview(&\$output, \$newsletter_id) [line 1437]

Function Parameters:

- **&\$output**
- **\$newsletter_id**

bool function newsletter_compose_show_testmail(&\$output, \$node_id, \$viewonly, \$edit_again, \$href, &\$edit_again) [line 1470]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which the content is connected
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$href** the action property of the HTML-form, the place where data will be POSTed
- **\$edit_again**

send an example mail of the current newsletter to the sender address

bool function newsletter_compose_toggle(\$node_id, \$article_id, \$select) [line 1215]

Function Parameters:

- *int* **\$node_id** identifies the newsletter
- *int* **\$article_id** is the article that needs to be (de)selected
- *bool* **\$select** TRUE=select article, FALSE=deselect article

select or deselect an article

this moves an existing article to the 'other' part of the list: selected articles will be added at the top of the unselected ones, unselected articles will be added at the bottom of the selected ones. we allow the sort order to show 'holes' (when an article in the middle of a list is (de)selected) because eventually all articles will be swallowed once a newsletter issue is published.

TRUE function newsletter_configuration_dialog_validate(&\$dialogdef) [line 732]

Function Parameters:

- *array* **&\$dialogdef** the dialog to check

validation of configuration dialog + rewriting the list of administrator email

validate dialog and rewrite admin email addresses as a side effect

array function newsletter_configuration_get_dialogdef(\$node_id, \$viewonly) [line 382]

Function Parameters:

- *int* **\$node_id** identifies the current newsletter
- *int* **\$viewonly** if TRUE the Save button is not displayed and config values cannot be changed

construct a dialog definition for the main newsletter configuration

- **Uses** \$USER, - \$CFG

bool function newsletter_configuration_save(&\$output, \$node_id, \$viewonly, &\$edit_again) [line 313]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which the content is connected
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise

validate and save the modified newsletter configuration linked to node \$node_id

this validates and saves the data that was submitted by the user.

If validation fails, the flag \$edit_again is set to TRUE and the return value is FALSE.

If the user has cancelled the operation, the flag \$edit_again is set to FALSE and the return value is also FALSE.

If the modified data is stored successfully, the return value is TRUE. The flag `edit_again` is set to TRUE if the user saved via the [Save] button, and to FALSE if the user saved via the [Done] button.

bool function newsletter_configuration_show(&\$output, \$node_id, \$module, \$viewonly, \$edit_again, \$href) [line 271]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

show a simple edit screen for the newsletter configuration

bool function newsletter_connect(&\$output, \$area_id, \$node_id, \$module) [line 85]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which we need to connect
- *array* **\$module** the module record straight from the database

connect this module to a node

this makes the link between the node `$node_id` in area `$area_id` and this module. In this case we create a single 'newsletters' record linked to `node_id` via `newsletter_id`.

bool function newsletter_content_show(&\$output, \$node_id, \$module) [line 220]

Function Parameters:

- *object* **&\$output** collects the html output (if any)

- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

show the main content screen (in fact documentation only)

this displays the simple overview screen with documentation. It ends with a cancel button that ends the whole edit operation for this node.

bool function newsletter_disconnect(&\$output, \$area_id, \$node_id, \$module) [line 54]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node from which we need to disconnect
- *array* **\$module** the module record straight from the database

disconnect this module from a node

this breaks the link between the node \$node_id in area \$area_id and this module. For now we simply delete the records in all tables 'newsletters*' using node_id == newsletter_id. Note the order in which the tables are emptied (to satisfy the FK constraints).

bool function newsletter_download_csv(&\$output, \$node_id, \$viewonly, &\$edit_again, \$href) [line 3081]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which the content is connected
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$href** the action property of the HTML-form, the place where data will be POSTed

send a CSV file of subscribers to the browser

CSV-rules are simple:

- first line shows column names
- any LF+CR, CR+LF or CR in data is replaced with a LF 0x0A
- any quote 0x22 in a quoted field is doubled
- the numeric field 'status' is not quoted, the other fields are
- fields are comma-separated (not TAB or semicolon)
- records are newline-separated (no CR's at all)

bool function newsletter_get_html(*\$newsletter_id*, *&\$toc*, *&\$title*, *&\$style*, *&\$body*, [*\$issue_id* = 0], [*\$announce* = FALSE]) [*line 4103*]

Function Parameters:

- *int* ***\$newsletter_id*** identifies the newsletter to construct
- *string* ***&\$toc*** receives newline delimited table of contents
- *string* ***&\$title*** receives the contents of the HTML title tag
- *string* ***&\$style*** receives the configured newsletter style information
- *string* ***&\$body*** receives the full body (but without the tag 'body')
- *int* ***\$issue_id*** is the unique issue ID for this issue (used for self reference)
- *bool* ***\$announce*** TRUE means: create only an announce message, FALSE=full HTML newsletter

construct the HTML-version of the newsletter in parts

this routine constructs the various parts of the HTML-version of the current newsletter. As an added bonus, this routine calculates a newline-delimited table of contents. This can be used to construct the newsletter archives lateron.

Note that the created #fragments start with 'h1' (and not h0). These continue to the end, including the colofon; the colofon is considered the N+1'th article. #toc leads to the start of the ToC, #top goes to the start of the body.

It is possible to have so-called 'ankeilers' (teasers) in announce messages. This works as follows: the Editor has to insert a horizontal line ('<hr>') into the article content somewhere. This implies that the announce message will not only contain the ToC of the newsletter but also will contain the title and the content of the articles upto the marker '<hr>'. This marker is removed from the announce article and an empty name anchor is inserted instead. Note that this only works for the FIRST '<hr>' in an article; any subsequent lines are left untouched. The named anchor is used as a target for the 'read more...' prompt that is added to the announce message after the ankeiler. This prompt is linked to the 'b' name in the full article online. If there is an ankeiler, there is also a link before it ('read online...') that links to the 'h' name in the full article online.

If none of the articles contains a horizontal line, we don't do ankeilers at all, but just the ToC (with links to the online articles) and the Colofon.

array function newsletter_get_html_templates(&\$config, \$newsletter_id, [\$issue_id = 0]) [line 4293]

Function Parameters:

- *array* **&\$config** contains newsletter configuration
- *int* **\$newsletter_id** aka node_id
- *int* **\$issue_id** is the unique issue ID for this issue (used for self reference)

construct the configurable part of the HTML-version of the newsletter, with parameters

this prepares the 'template'-fields in the HTML-version of the newsletter. These fields can contain the following parameters: {TITLE} - title of the newsletter

{SUBTITLE} - subtitle of the newsletter

{VOLUME} - issue volume number

{NUMBER} - issue number within volume

{DATIM} - yyyy-mm-dd hh:mm:ss is exact publication time stamp

{YEAR} - yyyy is [publication year (as in (C)Copyright {YEAR} {TITLE})]

{DATE} - yyyy-mm-dd is publication date

{TIME} - hh:mm is publication time

{ARCHIVE} - fully qualified link leading to newsletter archive

{CONTRIBUTION} - fully qualified link leading to visitor contribution dialog

{SUBSCRIBE} - fully qualified link leading to visitor subscription dialog

{UNSUBSCRIBE} - fully qualified link leading to visitor unsubscription dialog

{CONFIRM} - fully qualified link leading to visitor confirmation dialog (for completeness...)

{ISSUE} - fully qualified link leading to online version of newsletter

{PRINT} - fully qualified link leading to printversion of this issue (after publication)

Note that the links may yield an error, e.g. when the newsletter is configured to not allow contributions the link '{CONTRIBUTION}' will not actually work. It is the responsibility of the newsletter administrator to use the parameters in say the newsletter footer.

Also note that the '{PRINT}' link only really works when an issue is (finally) published. Before publication the issue number yields an error message; this is a feature (we don't want Smart Alec's to see the next issue before publication). When this routine is called from the COMPOSE-preview we don't know the issue_id yet, so we show a 0 (it is only a preview, right?). In the PUBLICATION-preview we use the correct issue_id (but that only works after publication). The same logic applies to {ISSUE}.

Finally, the print version opens in a (very clean) new window.

TRUE function newsletter_publish_dialog_validate(&\$dialogdef) [line 2150]

Function Parameters:

- *array* **&\$dialogdef** the dialog to check

validation of publication confirmation dialog checks checked checkbox

validate dialog and also make sure that the checkbox is checked

array function newsletter_publish_get_dialogdef(\$viewonly) [line 2120]

Function Parameters:

- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed

construct the dialog definition for confirmation of newsletter publication

bool function newsletter_publish_prepare_issue(&\$output, \$newsletter_id, &\$issue) [line 1804]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$newsletter_id** identifies the node/newsletter
- *array* **&\$issue** receives a copy of the issue-record that was saved

prepare the current newsletter for publication in the issues table

bool function newsletter_publish_save(&\$output, \$node_id, \$viewonly, &\$edit_again) [line 2182]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node from which we need to disconnect
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise

publish the prepared newsletter

this routine actually 'flips the bit' and officially publishes the prepared newsletter. Strategy: A: retrieve the key information about the upcoming issue (status is still NEWSLETTER_UNPUBLISHED B: determine which channel to use (sets status of issue) C: set status D: update next_number in configuration E: add issue+subscribers to the queue F: remove all selected articles now that they are part of the newsletter issue

If anything goes wrong during this process, we return FALSE and we won't be editing again, ie. the user is returned to the page manager. A little ugly but there's only a small chance that this fails. I hope.

bool function newsletter_publish_show(&\$output, \$node_id, \$module, \$viewonly, \$edit_again, \$href) [line 1727]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

show the newsletter publication confirmation screen for the current newsletter

bool function newsletter_publish_show_confirm(&\$output, \$node_id, \$viewonly, \$edit_again, \$href, &\$issue) [line 1944]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *\$array* **&\$issue** contains all data for the newsletter issue to be published

show a confirmation dialog for the publisher

the confirmation dialog requires the publisher to manually check the box for publishing; it is reset every time this dialog is created. We want to prevent accidental publications...

Note that we also check to see if the combination of volume and number is not used before. If it is, the publisher is not able to check the box and hence publish the newsletter. We don't want two different newsletters with the same volume/number identification.

void function newsletter_publish_show_preview(&\$output, \$newsletter_id) [line 2028]

Function Parameters:

- **&\$output**
- **\$newsletter_id**

bool function newsletter_publish_show_testmail(&\$output, \$node_id, \$viewonly, \$edit_again, \$href, &\$edit_again) [line 2061]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which the content is connected
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$href** the action property of the HTML-form, the place where data will be POSTed
- **\$edit_again**

send final example mail of the current newsletter from database to the publishers address

- **Uses \$USER**
- **Uses \$CFG**

array function newsletter_queue_get_dialogdef(\$viewonly) [line 4044]

Function Parameters:

- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed

construct an OK-button and a Cancel-button for the queue overview dialog

bool function newsletter_queue_save(&\$output, \$node_id, \$viewonly, &\$edit_again) [line 4006]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise

process a few entries in the queue (if any)

this routine is called whenever the user presses the [OK] button in the queue overview screen. This is a way to speed up emptying of the queue; we process a few pending messages each time.

bool function newsletter_queue_show(&\$output, \$node_id, \$module, \$viewonly, \$edit_again, \$href) [line 3908]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- **\$edit_again**

display a table with pending newsletter emails

this shows all entries in the queue as a table with these columns:

id | volume | number | next | errors | email | full_name

bool function newsletter_save(&\$output, \$area_id, \$node_id, \$module, \$viewonly, &\$edit_again, \$option) [line 189]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$option** the selected submenu option or NULL if user used the 'Content' option to get here

dispatcher for save routines

We use different subroutines for clarity.

bool function newsletter_show_edit(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href, \$option) [line 149]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *string* **\$option** the selected submenu option or NULL if user used the 'Content' option to get here

dispatcher for showing various dialog screens

there are a few dialogs that are accessed via this routine: - overview: a simple introduction to this module (via \$option==NULL) - configuration: all properties of the newsletter (main newsletters record) (\$option=configuration) - edit: add, edit, delete, rearrange and select articles (\$option=edit) - publish: approve/publish the prepared newsletter (\$option=publish) - subscriptions: manage subscriptions and blacklist (\$option=subscriptions) - queue: manage the mail queue (\$option=queue)

The actual work is done in subroutines.

void function newsletter_strip8859_1(\$text, 1) [line 4514]

Function Parameters:

- *string \$text* may contain chrs 128-255
- *string 1* all chars 128-255 transliterated to ascii equivalents

transliterate characters 128-255 to something 'ascii-like'

translate chars 128-255 to acceptable (?) ASCII used to make non-UTF-8 data behave (sort of) kludge, sorry :-(

array function newsletter_subscriptions_get_dialogdef(\$node_id, \$viewonly, \$subscriber_id) [line 2948]

Function Parameters:

- *int \$node_id* identifies the current newsletter
- *int \$viewonly* if TRUE the Save button is not displayed and values cannot be changed
- *int \$subscriber_id* identifies the subscriber, 0 for a new one

construct a dialog definition for a subscriber (add/edit)

this constructs a dialog definition which handles adding and editing subscribers In the case of adding a new subscriber we start with empty fields (mostly), otherwise we attempt to load the existing data from the DB.

If the subscriber_id is 0 we are adding a new subscriber.

array function newsletter_subscriptions_get_dialogdef_delete(\$viewonly) [line 3047]

Function Parameters:

- *int* **\$viewonly** if TRUE the Delete button is not displayed

construct a dialog definition for deletion of a subscriber

this constructs a dialog definition which handles deleting of a subscriber. There is not much to do: the subscriber_id to delete is conveyed via \$_GET[].

bool function newsletter_subscriptions_save(&\$output, \$node_id, \$viewonly, &\$edit_again, \$edit_again) [line 2794]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node alias the newsletter
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- **&\$edit_again**

handle saving & deleting of subscribers

this handles all actual subscriber add/edit/delete actions. criterion for delete is the existence of the delete-parameter (which identifies the subscriber to delete), the criterion for edit/add is the subscriber-parameter.

Part of the logic handling the [Cancel] button is done here too (see also [newsletter_subscriptions_show_edit\(\)](#)).

- **TODO** what to do with equal email addresses: can those simply be added or must they be unique?

bool function newsletter_subscriptions_show(&\$output, \$node_id, \$module, \$viewonly, &\$edit_again, \$href) [line 2321]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which the content is connected

- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$href** the action property of the HTML-form, the place where data will be POSTed

dispatcher for displaying newsletter subscriptions dialogs

bool function newsletter_subscriptions_show_add(&\$output, \$node_id, \$viewonly, \$edit_again, \$href) [line 2639]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *bool* **\$viewonly** if TRUE, adding/editing is not allowed (but simply showing the content is)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

show the dialog for adding a new subscriber

actual work is done in [newsletter_subscriptions_show_edit\(\)](#) but with subscriber_id set to 0 instead of a valid, existing subscriber_id.

bool function newsletter_subscriptions_show_delete(&\$output, \$node_id, \$subscriber_id, \$viewonly, \$edit_again, \$href) [line 2728]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *int* **\$subscriber_id** identifies the subscription to delete
- *bool* **\$viewonly** if TRUE, adding/editing is not allowed (but simply showing the content is)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

show a delete confirmation dialog

bool function newsletter_subscriptions_show_edit(&\$output, \$node_id, \$subscriber_id, \$viewonly, \$edit_again, \$href) [line 2676]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which this module is connected
- *int* **\$subscriber_id** identifies the subscription to edit (0 means add a new subscription dialog)
- *bool* **\$viewonly** if TRUE, adding/editing is not allowed (but simply showing the content is)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed

show dialog for editing (or adding) a subscriber

this displays the edit dialog for subscribers. It is also used by [newsletter_subscriptions_show_add\(\)](#) with subscriber_id = 0.

This dialog usually ends with three buttons: [Save] - saves the current subscriber and edits again [Done] - save the subscriber and return to the list of subscriptions [Cancel] - don't save and return to the list of subscriptions This poses a little problem when a new subscriber is added and the user clicks [Save]; there is no elegant way to convey the new subscriber_id (which was 0 in the add subscriber dialog) to the routine that subsequently edits the newly added subscriber. The (ugly) work around is to set the global \$newsletter_new_subscriber_id in the save routine and pick up the new value here.

Also part of the logic to deal with [Cancel] is handled here; the [Cancel]-button in this Add/Edit dialog (and the Delete dialog in [newsletter_subscriptions_show_delete\(\)](#) for that matter) should return to the list of subscriptions, whereas the [Cancel] button in the list of subscriptions should lead to the content overview in .

bool function newsletter_subscriptions_show_list(&\$output, \$node_id, \$viewonly, &\$edit_again, \$href) [line 2396]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which the content is connected
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed

- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$href** the action property of the HTML-form, the place where data will be POSTed

create the main newsletter subscriptions overview screen

Display a list of all current subscribers (if any)

Header

Explanation

Add a subscriber

ID Status Email Name

[D] id status email full_name

...

[D] id status email full_name

Download

Upload

[Cancel]

If there are no subscribers, only the Add + Upload links are visible (if editing is allowed). If there are subscribers, they are listed in a HTML-table. The [D]elete icon leads to the delete dialog but is only visible if editing is allowed, 'Add' leads to the add subscriber dialog (if editing is allowed), The ID field is clickable and leads to the edit dialog. The table headers are clickable and change the sort order. If there are many subscribers, the list is paginated.

The column with delete icons is only visible when `$viewonly == FALSE`.

Sort order: 1=ID, 2=Status,Email, 3=Email,ID, 4=Name,ID; negative=DESC; default +2

bool function newsletter_subscriptions_table(&\$output, \$newsletter_id, &\$records, \$viewonly, \$sort, \$limit, \$offset)
[line 2517]

Function Parameters:

- *object* **&\$output** collects the html output
- *int* **\$newsletter_id** aka the node_id
- *array* **&\$records** all subscribers that need to be displayed in the list
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *int* **\$sort** indicates sort order (1/id, 2/status, 3/email, 4/name), negative is descending

- *int* **\$limit**
 - of records per screen
- *in* **\$offset** offset in the database to the 1st record in the visible list

pretty-print a list of subscribers including sort options in the HTML table header

bool function newsletter_upload_csv(&\$output, \$node_id, \$viewonly, &\$edit_again, \$href) [line 3155]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$node_id** the node to which the content is connected
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$href** the action property of the HTML-form, the place where data will be POSTed

handle subscribers upload

this whole procedure works with 2 dialogs. The first dialog lets the user specify a file to upload and/or allows for a list of subscriber information (in CSV format). This dialog is presented here in [newsletter_upload_csv\(\)](#).

Once the CSV-records are submitted (file and/or text area), a second dialog is presented to the user (see [newsletter_upload_entries\(\)](#)). In that dialog the user can manipulate the status of the new subscribers but not the actual data.

In this routine we present the first dialog and we also validate the incoming data from that dialog. If that appears to be valid, we switch to phase two by calling [newsletter_upload_entries\(\)](#). If we are already in phase two, we continue with that routine immediately.

void function newsletter_upload_csv_extract(\$newsletter_id, &\$uploads, &\$row, &\$fields) [line 3464]

Function Parameters:

- *int* **\$newsletter_id** identifies the newsletter
- *array* **&\$uploads** receives processed upload data (4 fields)

- *array* **&\$row** contains the CSV-record, one field per array element
- *array* **&\$fields** contains the lists of CSV-fields to combine

helper routine to extract usable data from a single CSV record

this routine extracts the 0-based fields from \$row based on the information in the \$fields array. That array provides 4 lists of input fields that should be combined to form a single output field.

After constructing the four fields (and lowercasing the email address along the way), we take a look in the database to see if this newsletter already has this email address. If that is the case, we collect the subscriber_id and the current status in additional fields 4 and 5. If the email address does not yet exist, we simply set fields 4 and 5 to a 0, indicating a new subscriber.

The output fields are subsequently added to the \$uploads array.

array function newsletter_upload_csv_get_dialogdef(\$viewonly) [line 3202]

Function Parameters:

- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed

construct a dialog for specifying CSV upload

bool/array function newsletter_upload_csv_load(\$newsletter_id, &\$dialogdef) [line 3307]

Function Parameters:

- *int* **\$newsletter_id** identifies the newsletter
- *array* **&\$dialogdef** contains the valid results of the 1st upload dialog

massage the submitted data and return a neat array with subscriber information

bool function newsletter_upload_entries(&\$output, \$node_id, \$viewonly, &\$edit_again, \$href, &\$uploads) [line 3538]

Function Parameters:

- *object* **&\$output** collects the html output (if any)

- *int* **\$node_id** the node to which the content is connected
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$href** the action property of the HTML-form, the place where data will be POSTed
- *array* **&\$uploads** holds the preprocessed uploaded subscriber information (4 fields/row) or NULL

handle phase two of subscriber upload

After handling the CSV-upload in [newsletter_upload_csv\(\)](#), we call this routine to handle the actual entries that were uploaded.

The first time this routine is called, it is with a reference to all uploaded entries in \$uploads. This array is used to construct the initial dialog definition. Every time we come here after that the dialogdefinition is reconstructed based on the count parameter in the dialog, and the data is retrieved from \$_POST.

The main task here is to validate the data and subsequently show a neat HTML table with the uploaded entries. The user can then select the new status for the uploaded users. One of the options is to 'skip', i.e. prevent the uploaded entry from ending up in the database.

In order to make any errors a little more visible, we set the error flag in the status field even if the real error occurs in another field (e.g. the remarks field is too long). This way the status listbox turns red if there is an error.

Validation is handled 'manually' in two parts. The first part is all generic fields in the dialogdef, e.g. csrftoken. An error in those fields is fatal and will prevent saving of the data. The second part simply steps through the \$count entries that are stored in \$dialogdef. Errors in those entries are non-fatal, because the user can select 'skip' for erroneous entries and try again.

Note that we only take the non-skipped entries into account. If an entry contains errors, the user can (manually) set that record to 'skip' and that entry will not be used. Any errors in skipped entries are ignored when deciding if it is time to actually save the data.

Because we want an HTML table, we cannot use dialog_quickform(). Oh well.

void function newsletter_upload_entries_get_dialogdef(\$viewonly, \$edit_again, &\$uploads, 3) [line 3661]

Function Parameters:

- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *array* **&\$uploads** holds the preprocessed uploaded subscriber information (4 fields/row) or NULL

- *array* **3** dialog definition

construct a dialog definition, maybe filled with data from \$uploads

this constructs a dialogdefinition based on either the records in \$uploads OR from the \$_POST array. In the latter case the number of elements is determined by the counter in a hidden field.

There are a few generic fields: 'csrftoken','count','button_done','button_cancel' and there \$count sets of 'email','full_name','remarks','status','sid','old' with a 0-based index appended. Total size: \$count * 6 + 4 elements.

bool function newsletter_upload_entries_store(&\$output, &\$dialogdef, \$node_id) [line 3827]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *array* **&\$dialogdef** holds the complete upload dialog including data
- *int* **\$node_id** the node to which the content is connected

store the subscribers data in the database

this stores the contents of the dialogdef in the database. All data is already validated, it is simply a question of storing the data. The only twist is that we either update an existing record or insert a new one, depending on the 'sid' (subscriber_id) value: a 0 indicates that a new record is required.

The new status of an existing subscriber is calculated from both the old and the new value as follows: we simply use the maximum value of both. This means that

- 3 (blacklist) trumps everything: an import can never un-blacklist a subscriber. but selecting 'blacklist' in the upload overrules any existing value.
- 2 (approved): an approved subscriber stays approved, even if the upload would suggest only 'confirmed'. OTOH: if the subscriber is only 'confirmed', the upload can upgrade the subscriber to 'approved'.
- 1 (confirmed): any subscriber who is in the process of subscribing herself (status = 0/new) is automatically upgraded to 'confirmed' The other way around is not possible because the value 0 means 'skip' in the upload routine.
In other words: \$status = max(\$old,\$new). Easy.

string function newsletter_urls2cids(&\$mailer, \$html, &\$cids) [line 4380]

Function Parameters:

- *object* **&\$mailer** holds the object that constructs & sends the email message
- *string* **\$html** original HTML-string with href's and src's
- *array* **&\$cids** receives the list of replaced urls

search through \$html add selected files to email and replace url with cid

this routine attempts to store files in the email being currently constructed and subsequently replace familiar urls with a content-id generated by the Email object.

We do this for href="\$url", src="\$url" (html) and url(\$url) (css). There may be more urls to replace but those are less likely in a newsletter. Here is an example:

```
Example:  '<a title="foo" href="/path/to/file.php?file=/areas/exemplum/lagos.jpg" alt="lagos">'
$matches[0]:  '<a      title="foo"      href="/path/to/file.php?file=/areas/exemplum/lagos.jpg"
alt="lagos">'
$matches[1]:  '<a title="foo" '
$matches[2]:      'href'
$matches[3]:      '/path/to/file.php?file=/areas/exemplum/lagos.jpg'
$matches[4]:      '?file='
$matches[5]:      '/areas/exemplum/lagos.jpg'
$matches[6]:      ' alt="lagos">'
```

We want to replace \$matches[3] with something resembling 'cid:content-id@localhost' like so:
'' The example shows we need to catenate parts 1, 2, a quote, the \$cid, another quote and part 6.

We use \$mailer for storing attachments and we get a content_id in return.

- **TODO** we may be doing double work by searching/replacing the same url()'s again and again

void function newsletter_volume_number_available(\$newsletter_id, \$volume, \$number, 3) [line 4490]

Function Parameters:

- *int* **\$newsletter_id**

- *int* **\$volume**
- *int* **\$number**
- *bool* **3** TRUE if combination is available, FALSE means already exists or an error occurred

determine whether volume + number already exists in this newsletter

this check is performed while the preliminary newsletter issue is already in the issues table. Therefore there should be at most one record with that Volume + Number. If there are more, this combination is already taken.

require_once **dirname(__FILE__)."/newsletter_common.php"** [*line 39*]

newsletter_common.php

/program/modules/newsletter/newsletter_common.php - code shared between admin and view

This file defines various constants and subroutines used from both and [newsletter_view.php](#).

- **Package** wasmod_newsletter
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: newsletter_common.php,v 1.18 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
NEWSLETTER_ALERT_SITE = 2 [line 32]
NEWSLETTER_CHORE_ADD = add [line 51]
NEWSLETTER_CHORE_DELETE = delete [line 52]
NEWSLETTER_CHORE_DESELECT = deselect [line 53]
NEWSLETTER_CHORE_DOWN = down [line 56]
NEWSLETTER_CHORE_DOWNLOAD = download [line 61]
NEWSLETTER_CHORE_EDIT = edit [line 57]
NEWSLETTER_CHORE_LIST = list [line 60]
NEWSLETTER_CHORE_PREVIEW = preview [line 58]
NEWSLETTER_CHORE_SELECT = select [line 54]
NEWSLETTER_CHORE_TESTMAIL = testmail [line 59]
NEWSLETTER_CHORE_UP = up [line 55]
NEWSLETTER_CHORE_UPLOAD = upload [line 62]
NEWSLETTER_CONFIRM_ADD = A [line 64]
NEWSLETTER_CONFIRM_DEL = D [line 65]
NEWSLETTER_CONFIRM_LEN = 12 [line 66]
NEWSLETTER_CONTRIBUTION_POLICY_NO = 0 [line 35]
NEWSLETTER_CONTRIBUTION_POLICY_USERS = 1 [line 36]
NEWSLETTER_CONTRIBUTION_POLICY_YES = 2 [line 37]
NEWSLETTER_FULLMAIL_ONLY = 1 [line 31]
NEWSLETTER_FULLMAIL_SITE = 3 [line 33]
NEWSLETTER_SUBSCRIPTION_APPROVED = 2 [line 47]
NEWSLETTER_SUBSCRIPTION_BLACKLIST = 3 [line 48]
NEWSLETTER_SUBSCRIPTION_CONFIRMED = 1 [line 46]
NEWSLETTER_SUBSCRIPTION_NEW = 0 [line 45]
NEWSLETTER_SUBSCRIPTION_POLICY_NONE = 0 [line 39]
NEWSLETTER_SUBSCRIPTION_POLICY_ONE = 1 [line 40]
NEWSLETTER_SUBSCRIPTION_POLICY_TWO = 2 [line 41]
NEWSLETTER_SUBSCRIPTION_SKIP = 0 [line 44]
NEWSLETTER_UNPUBLISHED = 0 [line 30]
bool/array function newsletter_config($node_id) [line 96]
```

Function Parameters:

- *int* **\$node_id** the node to which the content is connected

retrieve the newsletter configuration

bool function newsletter_consolidate_subscribers(\$newsletter_id, \$subscriber_id, \$email, \$status) [line 181]

Function Parameters:

- *int* **\$newsletter_id** the newsletter of interest
- *int* **\$subscriber_id** the subscriber record we want to keep
- *string* **\$email** the corresponding email address
- *int* **\$status** the new status of the subscriber

remove obsolete subscriber data from the database

this removes superfluous subscribers from the database; eventually there should only be a single instance of an email address per newsletter. However, during the process of subscribing or re-subscribing it may be possible for different records with the same email address (in the same newsletter) to co-exist, be it with different values for status.

This routine is supposedly called after a particular record reaches the status of either NEWSLETTER_SUBSCRIPTION_APPROVED (2) or NEWSLETTER_SUBSCRIPTION_BLACKLIST (3). In those cases all other records with the same email address should simply be removed.

One difficulty may be that there can still be records in the queue that point to 'old' subscribers that are about to be deleted; that would break the foreign key. So, the strategy here is to

- change the current records in the queue so that they are linked to the new record (if the status is 'APPROVED')
- OR
- delete the current records in the queue if the status is 'BLACKLIST'.
- delete all other records with the same email address.

bool function newsletter_dialog_validate(&\$dialogdef) [line 125]

Function Parameters:

- *object* **&\$dialogdef** defines the dialog and will hold POSTed values

validate the data entered by the (un)subscriber

Apart from the standard validation we check for at least 1 @-sign in the email address. (It is non-trivial to validate an email address, it may not even be possible to do it in a single regular expression. See <http://stackoverflow.com/questions/201323> for more information.)

As an added bonus the email address is trimmed and standardised in lowercase.

Note that this routine is used both in [newsletter_view.php](#) where visitors can subscribe/unsubscribe themselves and where the administrator is responsible.

array function newsletter_get_emails(\$buffer) [*line 85*]

Function Parameters:

- *string* **\$buffer** holds the messy list of email addresses

create a neat array with email-addresses from a (possibly messy) text

this routine converts a (messy) list of email addresses to an array with a single address per array element. If there are no addresses, an empty array is returned. The addresses are supposed to be comma-delimited and/or newline and/or space delimited. Note that we expect only plain email addresses, no readable names etc. (KISS)

bool/int function newsletter_queue_run([\$node_id = NULL], [\$limit = 50]) [*line 285*]

Function Parameters:

- *int* **\$node_id** indicates which newsletter to handle
- *int* **\$limit** maximum number of mails to send this time

attempt to send out as many queued newsletters as possible

this constructs a list of all queued mails ordered by - issue_id - retry time (oldest go first; fifo) - queue_id

This attempts to send out as many queued newsletters as possible. Depending on the value of \$node_id we are looking at a specific newsletter (\$node_id == \$newsletter_id) or any newsletter (\$node_id = NULL). Either way we retrieve a list of at most \$limit records from the queue. We select only records that have a retry-time in the past, Within that selection, we order the records by issue_id and subsequently the retry time. This means that we can re-use the issue data if there are more queued requests for the same issue. In the same way we also

cache the newsletter configuration; read once and hopefully use many.

Note that we start each queue record with incrementing the error count before we do anything else; this means that if somehow the first record gets 'stuck' we can continue with the queue on the next run without hitting the same error over and over again.

This is a best effort; if there's an error we simply bail..

Note that this routine is not interactive (no output to a browser), so it can easily be called from a cronjob.

newsletter_cron.php

/program/modules/newsletter/newsletter_cron.php - interface to the cron-part of the newsletter module

This file defines the interface with the newsletter-module for cron. The interface consists of this function:

```
newsletter_cron()
```

This function is called whenever cron determines that it is time to perform this function.

- **Package** wasmod_newsletter
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: newsletter_cron.php,v 1.4 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function newsletter_cron() [line 48]

routine that is called periodically by cron

this routine performs household chores every now and then (interval: 15 minutes; see [newsletter_manifest.php](#)). An important chore is tending to any queued newsletters. Another one is getting rid of obsolete and unfinished new entries in newsletter_subscribers and existing entries with expired oncescodes.

`require_once dirname(__FILE__)."/newsletter_common.php"` *[line 36]*

newsletter_install.php

/program/modules/newsletter/newsletter_install.php - installer of the newsletter module

This file contains the newsletter module installer. The interface consists of these functions:

```
newsletter_install(&$messages,$module_id)
newsletter_upgrade(&$messages,$module_id)
newsletter_uninstall(&$messages,$module_id)
newsletter_demodata(&$messages,$module_id,&$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wasmod_newsletter
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: newsletter_install.php,v 1.8 2016/06/16 13:48:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function newsletter_demodata(&\$messages, \$module_id, &\$config, \$manifest) [line 148]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table
- *array* **&\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this module

add demonstration data to the system

this routine adds demodata for this module to the database.

Note that a routine by this name should exist (even if it does nothing).

Note If the module is installed via the Install Wizard, this routine is called. However, if a module is installed as an additional module after installation, the `{ $module }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

The array `$config` contains the following information.

```
$config['language_key'] => install language code (eg. 'en')
$config['dir']          => path to CMS Root Directory (eg. /home/httpd/htdocs)
$config['www']          => URL of CMS Root Directory (eg. http://exemplum.eu)
$config['progdir']      => path to program directory (eg. /home/httpd/htdocs/program)
$config['progwww']      => URL of program directory (eg. http://exemplum.eu/program)
$config['datadir']      => path to data directory (eg. /home/httpd/wasdata/alb2c3d4e5f6)
$config['user_username'] => userid of webmaster (eg. wblader)
$config['user_full_name'] => full name of webmaster (eg. Wilhelmina Bladergroen)
$config['user_email']   => email of webmaster (eg. w.bladergroen@exemplum.eu)
$config['user_id']      => numerical user_id (usually 1)
$config['demo_salt']    => password salt for all demodata accounts
$config['demo_password'] => password for all demodata accounts
$config['demo_areas']   => array with demo area data
$config['demo_groups']  => array with demo group data
$config['demo_users']   => array with demo user data
$config['demo_nodes']   => array with demo node data
$config['demo_string']  => array with demo strings from /program/install/languages/LL/demodata.php
$config['demo_replace'] => array with search/replace pairs to 'jazz up' the demo strings
```

With this information, we can add a demonstration configuration for the public area, which shows off the possibilities. Note that we add our own additions to the array `$config` so other modules and themes can determine the correct status quo w.r.t. the demodata nodes etc.

- **TODO** There are various stubs that need to be fixed.

bool function `newsletter_install(&$messages, $module_id)` [line 61]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

install the module

this routine installs the module. For this module there is nothing to install, so we simply return success.

Note that the record for this module is already created in the modules table; the pkey is \$module_id.

bool function newsletter_uninstall(&\$messages, \$module_id) [line 92]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

uninstall the module

this is a hook for future extensions of Website@School. For now we simply return success.

bool function newsletter_upgrade(&\$messages, \$module_id) [line 78]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

upgrade the module

this routine performs an upgrade to the installed module. Note that the newsletter module does not need any upgrade at all because there never was an earlier version.

For now this routine is a nop.

newsletter_manifest.php

/program/modules/newsletter/newsletter_manifest.php - description of the newsletter module

This file defines the various components of the newsletter module such as the names of the various scripts and version information. This file is used when this module is installed.

- **Package** wasmod_newsletter
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: newsletter_manifest.php,v 1.5 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

newsletter_search.php

/program/modules/newsletter/newsletter_search.php - interface to the search-part of this module

This file defines the interface with this module for searching content. The interface consists of a (temporary) table and this function:

```
bool newsletter_search(&$hits,&$results,$qwords, $limit, $offset)
```

The function is called whenever module content is to be searched. At that point the helper table, which is called 'search_nodes', exists and is populated.

- **Package** wasmod_newsletter
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: newsletter_search.php,v 1.5 2016/04/05 06:34:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool|array function newsletter_search(&\$hits, &\$results, \$qwords, \$limit, [\$offset = 0]) [line 59]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results

search this module's content in selected nodes for keywords in \$qwords

This interface is documented in [htmlpage_search\(\)](#) in [htmlpage_search.php](#).

We have to search the following fields in two tables:

a. newsletters: 1 record/newsletter: title/subtitle/header/introduction/footer

b. newsletter_issues: 0,1,...N records/newsletter: html_title/html_body

Newsletters and -issues qualify only when the channel/status is 2 (website) or 3 (both) Hits in newsletters link to the main page (node=\$node_id) Hits in the newsletter_issues link to node=\$node_id,volume=\$volume,number=\$number

bool/int function newsletter_search_results(\$table, \$where) [*line 165*]

Function Parameters:

- *string* **\$table** name of table to search
- *string* **\$where** where clause to select hits in \$table

count the number of hits in \$table

newsletter_view.php

/program/modules/newsletter/newsletter_view.php - interface to the view-part of the newsletter module

This file defines the interface with the newsletter-module for viewing content. The interface consists of this function:

```
newsletter_view(&$output,$area_id,$node_id,$module)
```

This function is called from /index.php when the node to display is connected to this module.

- **Package** wasmod_newsletter
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: newsletter_view.php,v 1.13 2016/05/26 13:22:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
NEWSLETTER_DIRNAME = dirname(__FILE__) [line 37]  
NEWSLETTER_REFERENCE = sha1(__FILE__."":__LINE__) [line 40]  
bool function newsletter_view(&$theme, $area_id, $node_id, $module) [line 74]
```

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

display the content of the newsletter linked to node \$node_id

this routine is basically a dispatcher. First we decide which policies are in place for this visitor (who might be a user):

- **flag_a**: the newsletter archive is available
- **flag_c**: the visitor is allowed to contribute
- **flag_s**: the visitor is allowed to subscribe/unsubscribe

After that we decide which routine to perform based on the parameters that are specified. The following combinations are recognised:

- archive: volume=v and number=n; shows 0, 1 or more old newsletters
- contribution: contribution=c: handle a visitor contribution
- subscribe: subscribe=s or unsubscribe=u: handle subscriptions

If none of those are applicable, the default page is displayed, possibly with the latest newsletter (if policy allows it).

The actual work is done in subroutines. The values of these flags are added to the config record which is passed to the subroutines.

bool function newsletter_view_add_navbar(&\$theme, \$config, [\$issue_id = NULL]) [line 172]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$config** newsletter configuration + a few policy-flags
- *int* **\$issue_id** identify issue to show 'stand-alone' or NULL for none

add a navigation bar to the output

if the corresponding policy allows this creates links to the archive, the contribution dialog and the subscribe/unsubscribe logic. We also (optionally) show a link to display the specified issue 'stand-alone' (in a new window).

bool function newsletter_view_archive(&\$theme, \$config, \$node_id, \$volume, \$number) [line 260]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$config** newsletter configuration + a few policy-flags
- *int* **\$node_id** the node to which this module is connected
- *int* **\$volume** is the volume number > 0 or a wildcard (<= 0)
- *int* **\$number** is the issue number > 0 or a wildcard (<= 0)

display two levels of newsletter issues: volumes+number or numbers+toc

if \$volume == 0 we show all available volumes with clickable links to the numbers (including a print link) as nested lists. If \$volume == V we show all numbers in volume V

(+print link) AND the clickable ToC of every number. We keep everything in at most two nested lists to prevent cluttering.

bool function newsletter_view_confirm(&\$theme, \$config, \$node_id, [\$code = 1]) [line 571]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$config** newsletter configuration + a few policy-flags
- *int* **\$node_id** the node to which this module is connected
- **\$code**

handle the confirmation code for subscribing/unsubscribing

array function newsletter_view_confirm_get_dialogdef() [line 619]

construct a dialog definition for confirm code

this constructs a dialog definition which presents the user with a 1-field dialog where she can enter the oncescode that was emailed to her earlier.

bool function newsletter_view_confirm_process_code(\$theme, \$config, \$code, &\$theme) [line 650]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$config** newsletter configuration + a few policy-flags
- *string* **\$code** the oncescode entered by the user
- **\$theme**

process a code confirming a subscribe or unsubscribe action

we arrive here with a code that has the action to perform embedded. It consists of a quasi-random string, the action to perform ('d' means unsubscribe, 'a' means subscribe) and finally a uniquemaker based on a subscriber_id.

bool function newsletter_view_contribution(&\$theme, \$config, \$node_id, \$value) [line 987]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$config** newsletter configuration + a few policy-flags
- *int* **\$node_id** the node to which this module is connected
- *int* **\$value** indicates the phase in the contribution process

handle user contribution of articles

this eventually yields a user contribution in the form of a text-only article added to the list of UNselected articles in the database. we use the logic from tokenlib as we did in [mailpage_view.php](#). Data entry is a two-step process: one screen with editable text and a preview-button, another screen with uneditable text and a submit-button This makes it impossible to simply keep submitting a story; the first screen provides a token, and the second screen only works after a minimum delay.

void function newsletter_view_contribution_form(&\$theme, \$config, \$dialogdef) [*line 1172*]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$dialogdef** array that defines the input fields
- *array* **\$config** mnewsletter configuration data

display the contribution form

this displays the contribution form.

array function newsletter_view_contribution_get_dialogdef(\$config, \$token_key) [*line 1088*]

Function Parameters:

- *array* **\$config** newsletter configuration
- *string* **\$token_key**

construct a dialog definition for the visitor's contribution form

void function newsletter_view_contribution_preview(&\$theme, \$config, \$dialogdef, \$ip_addr) [line 1198]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$dialogdef** array that defines the input fields
- *string* **\$ip_addr** the originating IP-address
- *array* **\$config** newsletter configuration data

show a preview of the message to the visitor

this shows a preview of the article to the visitor. Nothing is editable, it is view-only. The only option is to either press the Send-button to actually submit the message OR to press the Edit button to go back to the editable form.

Sending a message is a two-step procedure by design.

bool function newsletter_view_contribution_send(&\$theme, \$config, \$dialogdef, \$ip_addr, \$delay) [line 1250]

Function Parameters:

- *object* **&\$theme** collects the (html) output and provides queue_alert()
- *array* **\$config** mailpage configuration data in a (nested) array
- *array* **\$dialogdef** array that defines the data fields including values
- *string* **\$ip_addr** the originating IP-address
- *int* **\$delay** the # of seconds since time=t0

add the contribution to the list of unselected articles and mail the admins

In order to get a feeling for the time a visitor needs, we also record the delay (in seconds) next to the visitor's IP address.

- **TODO** extra validation of set_mailreplyto and set_subject?
- **TODO** more available parameters in subject_line?
- **TODO** make body of mail configurable?

void function newsletter_view_contribution_thankyou(&\$theme, \$config, \$dialogdef, \$ip_addr) [line 1343]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$dialogdef** array that defines the input fields
- *string* **\$ip_addr** the originating IP-address
- *array* **\$config** mailpage configuration data in a (nested) array

thank the visitor for the article and show a text copy too

Almost the same as {*@see* newsletter_view_contribution_preview()}.

- **TODO** should we have an OK button at all???

bool function newsletter_view_home(&\$theme, \$config, \$node_id) [line 129]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$config** newsletter configuration + a few policy-flags
- *int* **\$node_id** the node to which this module is connected

display the default page, possibly with the latest newsletter embedded

bool function newsletter_view_issue(&\$theme, \$config, \$node_id, \$issue_id) [line 223]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$config** newsletter configuration + a few policy-flags

- *int* **\$node_id** the node to which this module is connected
- *int* **\$issue_id** explicit issue to show 'stand-alone'

show a 'stand-alone' version of the newsletter

this is almost the same routine as [newsletter_publish_show_preview\(\)](#) and [newsletter_compose_show_preview\(\)](#). The difference is that we look for a specific issue in the archives (could be the latest issue). However, the issue must be in the archive as indicated by the status. If this issue was sent by mail only, we exit with fatal error 099. We assume we have our own window (arrived here via target="_blank").

bool function newsletter_view_subscribe(&\$theme, \$config, \$node_id) [line 777]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$config** newsletter configuration + a few policy-flags
- *int* **\$node_id** the node to which this module is connected

show and process a subscription request (phase 1)

array function newsletter_view_subscribe_get_dialogdef(\$show_remarks) [line 838]

Function Parameters:

- *bool* **\$show_remarks** if TRUE we show the remarks field too

construct a dialog definition for subscribing

this constructs a dialog definition which presents the user with a 2 or 3 field dialog where she can enter her email-address, full name and perhaps additional remarks to subscribe.

bool/string function newsletter_view_subscribe_send_code(&\$theme, \$config, \$email, \$full_name, \$remarks) [line 886]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$config** newsletter configuration + a few policy-flags

- *string* **\$email** destination email address
- **\$full_name**
- **\$remarks**

send a quasi-random subscribe confirmation code to \$email

bool function newsletter_view_unsubscribe(&\$theme, \$config, \$node_id) [line 393]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$config** newsletter configuration + a few policy-flags
- *int* **\$node_id** the node to which this module is connected

handle unsubscriptions (phase 1)

this routine shows the dialog where the subscriber can enter her email address to unsubscribe OR processes the same dialog. In the latter case, if the oncescode was sent successfully, the phase 2 dialg (confirm) is displayed.

The email that is sent contains the oncescode which can be entered in that dialog. The email also contains a convenient clickable link.

array function newsletter_view_unsubscribe_get_dialogdef() [line 449]

construct a dialog definition for unsubscribing

this constructs a dialog definition which presents the user with a 1-field dialog where she can enter her email-address (to unsubscribe).

bool/string function newsletter_view_unsubscribe_send_code(&\$theme, \$config, \$email) [line 474]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$config** newsletter configuration + a few policy-flags
- *string* **\$email** destination email address

send a quasi-random unsubscribe confirmation code to \$email

```
require_once NEWSLETTER_DIRNAME."/newsletter_common.php" [line 38]
```

```
require_once $CFG->progdire."/lib/tokenlib.php" [line 41]
```


Package wasmod_redirect Procedural Elements

redirect.php

/program/modules/redirect/languages/ar/redirect.php

Language: ar (Ø§Ù,,Ø¹Ø±Ø¨ÙŠØ©) Release: 0.90.3 / 2012041700 (2012-04-17)

- **Package** wasmod_redirect
- **Author** Said Taki < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.4 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/da/redirect.php

Language: da (Dansk) Release: 0.90.3 / 2012041700 (2012-04-17)

- **Package** wasmod_redirect
- **Author** Christian Borum Loebner - Olesen < translators@websiteatschool.eu >
- **Version** \$Id: redirect.php,v 1.4 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/en/redirect.php - translated messages for module (English)

- **Package** wasmod_redirect
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: redirect.php,v 1.7 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/es/redirect.php

Language: es (Español) Release: 0.90.3 / 2012041700 (2012-04-17)

- **Package** wasmod_redirect
- **Author** Anouk Coumans < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.4 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/fa/redirect.php

Language: fa (â€«Ù•Ø§Ø±Ø³Ùœ) Release: 0.90.3 / 2012041700 (2012-04-17)

- **Package** wasmod_redirect
- **Author** A. Darvishi < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.4 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/hu/redirect.php

Language: hu (Magyar) Release: 0.90.3 / 2012041700 (2012-04-17)

- **Package** wasmod_redirect
- **Author** Gergely Sipos, Erika Swiderski < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.4 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/nl/redirect.php - translated messages for module (Dutch)

- **Package** wasmod_redirect
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: redirect.php,v 1.5 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/pl/redirect.php

Language: pl (Polski) Release: 0.90.3 / 2012041700 (2012-04-17)

- **Package** wasmod_redirect
- **Author** Waldemar Pankiw < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.4 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/tr/redirect.php

Language: tr (Türkçe) Release: 0.90.3 / 2012041700 (2012-04-17)

- **Package** wasmod_redirect
- **Author** Özgür Gaga translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.4 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect.php

/program/modules/redirect/languages/zh/redirect.php

Language: zh (ä, -æ-†) Release: 0.90.3 / 2012041700 (2012-04-17)

- **Package** wasmod_redirect
- **Author** Liu Jing Fang < translators@websiteatschool.eu>
- **Version** \$Id: redirect.php,v 1.4 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Vereniging Website At School, Amsterdam
- **License** [GNU AGPLv3+Additional Terms](#)

redirect_admin.php

/program/modules/redirect/redirect_admin.php - management interface for redirect-module

This file defines the administrative interface to this module. The interface consists of the following four functions.

```
redirect_disconnect(&$output,$area_id,$node_id,$module)
redirect_connect(&$output,$area_id,$node_id,$module)
redirect_show_edit(&$output,$area_id,$node_id,$module,$viewonly,$edit_again,$href,$option)
redirect_save(&$output,$area_id,$node_id,$module,$viewonly,&$edit_again,$option)
```

These functions are called from pagemanagerlib.php whenever necessary.

- **Package** wasmod_redirect
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: redirect_admin.php,v 1.6 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function redirect_connect(&\$output, \$area_id, \$node_id, \$module) [line 77]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which we need to connect
- *array* **\$module** the module record straight from the database

connect this module to a node

this makes the link between the node \$node_id in area \$area_id and this module. Since all this module does is manipulating the node record, this is a no-op.

bool function redirect_disconnect(&\$output, \$area_id, \$node_id, \$module) [line 52]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node from which we need to disconnect
- *array* **\$module** the module record straight from the database

disconnect this module from a node

this breaks the link between the node \$node_id in area \$area_id and this module. For now we simply delete the content of the link_url and link_target fields in the node record.

array function redirect_get_dialogdef(\$viewonly) [line 224]

Function Parameters:

- *int* **\$viewonly** if TRUE the Save button is not displayed and values cannot be changed by the user

construct a dialog definition for the redirect value

bool function redirect_save(&\$output, \$area_id, \$node_id, \$module, \$viewonly, &\$edit_again) [line 177]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise

save the modified content data of this module linked to node \$node_id

this validates and saves the data that was submitted by the user. If validation fails, or storing the data doesn't work, the flag \$edit_again is set to TRUE and the return value is FALSE.

If the user has cancelled the operation, the flag \$edit_again is set to FALSE and the return value is also FALSE.

If the modified data is stored successfully, the return value is TRUE. The value of \$edit_again is determined by the key used to save: [Save] brings the user back to edit (\$edit_again = TRUE) where [Done] leaves editing altogether (\$edit_again = FALSE).

Here is a summary of return values.

\$retval	\$edit_again	Action
-----	-----	-----
FALSE	FALSE	[Cancel] Stop editing, unlock & return to tree view
FALSE	TRUE	[Save] or [Done] failed: redo the dialog and let user correct mistakes
TRUE	FALSE	[Done] pressed, everything saved succesfully, we're done
TRUE	TRUE	[Save] pressed, everything saved succesfully but continue editing

The parameter \$option is not used in this module.

bool function redirect_show_edit(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href, \$option)
[line 111]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

present the user with a dialog to modify the redirect that is connected to node \$node_id

this prepares a dialog for the user filled with existing data (if any), possibly allowing the user to modify the content. If the flag \$viewonly is TRUE, this routine should only display the content rather than let the user edit it. If the flag \$edit_again is TRUE, the routine should use the data available in the \$_POST array, otherwise it should read the data from the database (or wherever the data comes from). The parameter \$href is the place where the form should be POST'ed.

The dialog should be added to the \$output object. Useful routines are:

```
$output->add_content($content): add $content to the content area  
$output->add_message($message): add $message to the message area (feedback to the user)  
$output->add_popup_bottom($message): popup $message in browser after loading the page (uses javascript)  
$output->add_popup_top($message): popup $message in browser before loading the page (uses javascript)
```

The parameter \$option is not used in this module.

redirect_cron.php

/program/modules/redirect/redirect_cron.php - interface to the cron-part of the redirect module

This file defines the interface with the redirect-module for cron. The interface consists of this function:

```
redirect_cron()
```

This function is called whenever cron determines that it is time to perform this function.

- **Package** wasmod_redirect
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: redirect_cron.php,v 1.5 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function redirect_cron() [*line 44*]

routine that is called periodically by cron

there is nothing in this module that needs to be done by cron, so this routine is basically a nop.

redirect_install.php

/program/modules/redirect/redirect_install.php - installer of the redirect module

This file contains the redirect module installer. The interface consists of these functions:

```
redirect_install(&$messages,$module_id)
redirect_upgrade(&$messages,$module_id)
redirect_uninstall(&$messages,$module_id)
redirect_demodata(&$messages,$module_id,&$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wasmod_redirect
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: redirect_install.php,v 1.5 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function redirect_demodata(&\$messages, \$module_id, &\$config, \$manifest) [line 122]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table
- *array* **&\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this module

add demonstration data to the system

this routine is a no-op because there is no redirect demodata and if it is there, it is already created in the main demodata-routine in /program/install/demodata.php.

This routine is retained here as an example and also because a routine by this name should exist (even if it does nothing).

Note If the module is installed via the Install Wizard, this routine is called. However, if a module is installed as an additional module after installation, the `{ $mode }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

bool function redirect_install(&\$messages, \$module_id) [line 61]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

install the module

this routine installs the module. For this module there is nothing to install, so we simply return success.

Note that the record for this module is already created in the modules table; the pkey is `$module_id`.

bool function redirect_uninstall(&\$messages, \$module_id) [line 92]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

uninstall the module

this is a hook for future extensions of Website@School. For now we simply return success.

bool function redirect_upgrade(&\$messages, \$module_id) [line 78]

Function Parameters:

- *array* **&\$messages** collects the (error) messages

- *int* **\$module_id** the key for this module in the modules table

upgrade the module

this routine performs an upgrade to the installed module. Note that this minimalistic 'redirect' module does not need any upgrade at all because there never was an earlier version.

For now this routine is a nop.

redirect_manifest.php

/program/modules/redirect/redirect_manifest.php - description of the redirect module

This file defines the various components of the redirect module such as the names of the various scripts and version information. This file is used when this module is installed.

- **Package** wasmod_redirect
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: redirect_manifest.php,v 1.8 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

redirect_search.php

/program/modules/redirect/redirect_search.php - interface to the search-part of this module

This file defines the interface with this module for searching content. The interface consists of a (temporary) table and this function:

```
bool redirect_search(&$hits,&$results,$qwords, $limit, $offset)
```

The function is called whenever module content is to be searched. At that point the helper table, which is called 'search_nodes', exists and is populated.

- **Package** wasmod_redirect
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: redirect_search.php,v 1.5 2016/04/01 18:31:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool|array function redirect_search(&\$hits, &\$results, \$qwords, \$limit, [\$offset = 0]) [*line 49*]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results

search this module's content in selected nodes for keywords in \$qwords

This interface is documented in [htmlpage_search\(\)](#) in [htmlpage_search.php](#).

redirect_view.php

/program/modules/redirect/redirect_view.php - interface to the view-part of the redirect module

This file defines the interface with the redirect-module for viewing content. The interface consists of this function:

```
redirect_view(&$output,$area_id,$node_id,$module)
```

This function is called from /index.php when the node to display is connected to this module.

However, since this module is the redirect module, this routine is never called because the execution of the redirection is done in main_index.php rather than here (deeply nested) in a module. At that higher level we are in a better position to handle the premature end of execution of main_index.php following sending the redirect header. Therefore this routine, too, is a nop.

- **Package** wasmod_redirect
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: redirect_view.php,v 1.4 2016/03/23 09:28:24 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

```
bool function redirect_view(&$theme, $area_id, $node_id, $module) [line 62]
```

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

display the content of the redirect linked to node \$node_id

Since this module is the redirect module, this routine should never be called because the execution of the redirection is done in main_index.php rather than here (deeply nested) in a module. At that higher level we are in a better position to handle the premature end of execution of main_index.php following sending the redirect header. Therefore this routine is a

nop.

However, it is possible that the module is (still) connected to the node but without a (valid) url. In that case the visitor sees a blank page but with complete navigation etc. Can't win 'm all...

Package wasmod_search Procedural Elements

search_tabledefs.php

/program/modules/search/install/search_tabledefs.php - data definition for module

One table is defined: search. It contains the configuration data for a search (header, introduction, etc.)

This module also uses a few temporary tables. These are created on the fly and eventually lead to a table that contains the nodes that are to be searched. The structure of that table looks like this:

```
CREATE TABLE search_nodes (node_id int(11) NOT NULL DEFAULT '0' PRIMARY KEY (node_id))
```

It is created in search_construct_nodelist() and used in all modules, see [search_module\(\)](#).

- **Package** wasmod_search
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: search_tabledefs.php,v 1.1 2016/04/01 18:31:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

search.php

/program/modules/search/languages/en/search.php - translated messages for module (English)

- **Package** wasmod_search
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: search.php,v 1.1 2016/04/01 18:31:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

search.php

/program/modules/search/languages/nl/search.php - translated messages for module (Dutch)

- **Package** wasmod_search
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: search.php,v 1.1 2016/04/01 18:31:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

search_admin.php

/program/modules/search/search_admin.php - management interface for search-module

This file defines the administrative interface to this module. The interface consists of the following four functions.

```
search_disconnect(&$output,$area_id,$node_id,$module)
search_connect(&$output,$area_id,$node_id,$module)
search_show_edit(&$output,$area_id,$node_id,$module,$viewonly,$edit_again,$href,$option)
search_save(&$output,$area_id,$node_id,$module,$viewonly,&$edit_again,$option)
```

These functions are called from pagemanagerlib.php whenever necessary.

- **Package** wasmod_search
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: search_admin.php,v 1.1 2016/04/01 18:31:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function search_connect(&\$output, \$area_id, \$node_id, \$module) [*line 71*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which we need to connect
- *array* **\$module** the module record straight from the database

connect this module to a node

this makes the link between the node \$node_id in area \$area_id and this module. In this case we create a single 'search' record linked to node_id.

bool function search_disconnect(&\$output, \$area_id, \$node_id, \$module) [*line 53*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node from which we need to disconnect
- *array* **\$module** the module record straight from the database

disconnect this module from a node

this breaks the link between the node \$node_id in area \$area_id and this module. For now we simply delete the record in 'search'. Note that the db_delete() here is a simple best effort: if it fails we do nothing but return FALSE;

array function search_get_dialogdef(\$viewonly) [line 240]

Function Parameters:

- *int* **\$viewonly** if TRUE the Save button is not displayed and values cannot be changed by the user

construct a dialog definition for the search 'scope' and other parameters

bool function search_save(&\$output, \$area_id, \$node_id, \$module, \$viewonly, &\$edit_again, \$option) [line 190]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

save the modified content data of this module linked to node \$node_id

this validates and saves the data that was submitted by the user. If validation fails, or storing the data doesn't work, the flag \$edit_again is set to TRUE and the return value is FALSE.

If the user has cancelled the operation, the flag \$edit_again is set to FALSE and the return value is also FALSE.

If the modified data is stored successfully, the return value is TRUE and the value of \$edit_again is based on the button used to save ([Save] or [Done]).

Here is a summary of return values.

\$retval	\$edit_again	Action
-----	-----	-----
FALSE	FALSE	[Cancel] Stop editing, unlock & return to tree view
FALSE	TRUE	[Save] or [Done] failed: redo the dialog and let user correct mistakes
TRUE	FALSE	[Done] pressed, everything saved succesfully, we're done
TRUE	TRUE	[Save] pressed, everything saved succesfully but continue editing

The parameter \$option is not used in this module.

bool function search_show_edit(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href, \$option)
[line 125]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

present the user with a dialog to modify the search that is connected to node \$node_id

this prepares a dialog for the user filled with existing data (if any), possibly allowing the user to modify the content. If the flag \$viewonly is TRUE, this routine should only display the content rather than let the user edit it. If the flag \$edit_again is TRUE, the routine should use the data available in the \$_POST array, otherwise it should read the data from the database

(or wherever the data comes from). The parameter \$href is the place where the form should be POST'ed.

The dialog should be added to the \$output object. Useful routines are:

```
$output->add_content($content): add $content to the content area  
$output->add_message($message): add $message to the message area (feedback to the user)  
$output->add_popup_bottom($message): popup $message in browser after loading the page (uses javascript)  
$output->add_popup_top($message): popup $message in browser before loading the page (uses javascript)
```

The parameter \$option is not used in this module.

search_cron.php

/program/modules/search/search_cron.php - interface to the cron-part of the search module

This file defines the interface with the search-module for cron. The interface consists of this function:

```
search_cron( )
```

This function is called whenever cron determines that it is time to perform this function.

- **Package** wasmod_search
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: search_cron.php,v 1.1 2016/04/01 18:31:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function search_cron() [*line 44*]

routine that is called periodically by cron

there is nothing in this module that needs to be done by cron, so this routine is basically a nop.

search_install.php

/program/modules/search/search_install.php - installer of the search module

This file contains the search module installer. The interface consists of these functions:

```
search_install(&$messages,$module_id)
search_upgrade(&$messages,$module_id)
search_uninstall(&$messages,$module_id)
search_demodata(&$messages,$module_id,&$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wasmod_search
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: search_install.php,v 1.1 2016/04/01 18:31:50 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function search_demodata(&\$messages, \$module_id, &\$config, \$manifest) [*line 119*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table
- *array* **&\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this module

add demonstration data to the system

this routine is a no-op because there is no search demodata and if it is there, it is already created in the main demodata-routine in `/program/install/demodata.php`.

This routine is retained here as an example and also because a routine by this name should exist (even if it does nothing).

Note If the module is installed via the Install Wizard, this routine is called. However, if a module is installed as an additional module after installation, the `{ $mode }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

bool function search_install(&\$messages, \$module_id) [line 61]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

install the module

this routine installs the module. For this module there is nothing to install, so we simply return success.

Note that the record for this module is already created in the modules table; the pkey is `$module_id`.

bool function search_uninstall(&\$messages, \$module_id) [line 89]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

uninstall the module

this is a hook for future extensions of Website@School. For now we simply return success.

bool function search_upgrade(&\$messages, \$module_id) [line 75]

Function Parameters:

- *array* **&\$messages** collects the (error) messages

- *int* **\$module_id** the key for this module in the modules table

upgrade the module

this routine performs an upgrade to the installed module. However, since there was no earlier version, this is a nop.

search_manifest.php

/program/modules/search/search_manifest.php - description of the search module

This file defines the various components of the search module such as the names of the various scripts and version information. This file is used when this module is installed.

- **Package** wasmod_search
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: search_manifest.php,v 1.2 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

search_search.php

/program/modules/search/search_search.php - interface to the search-part of this module

This file defines the interface with this module for searching content. The interface consists of a (temporary) table and this function:

```
bool search_search(&$hits,&$results,$qwords, $limit, $offset)
```

The function is called whenever module content is to be searched. At that point the helper table, which is called 'search_nodes', exists and is populated.

- **Package** wasmod_search
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: search_search.php,v 1.2 2016/04/05 14:33:01 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool|array function search_search(&\$hits, &\$results, \$qwords, \$limit, [\$offset = 0]) [line 49]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results

search this module's content in selected nodes for keywords in \$qwords

This interface is documented in [htmlpage_search\(\)](#) in [htmlpage_search.php](#).

search_view.php

/program/modules/search/search_view.php - interface to the view-part of the search module

This file defines the interface with the search-module for viewing content. The interface consists of this function:

```
search_view(&$output,$area_id,$node_id,$module)
```

This function is called from /index.php when the node to display is connected to this module.

- **Package** wasmod_search
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: search_view.php,v 1.6 2016/04/07 12:06:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

output function search(&\$theme, \$config, \$qwords, \$q, \$r, \$o, \$t) [line 251]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$config** search configuration data
- *array* **\$qwords** list of 1 or more keywords to search for (original AND lowercase each)
- *string* **\$q** original unprocessed query
- *int* **\$r** results per page
- *int* **\$o** offset
- *string* **\$t** comma separated list with hints about # of hits in various modules

perform an actual search

At this point we have collected the following parameters:

- string \$q: the question, not empty
- int \$r: # of results per page
- int \$o: offset relative to the very first result
- string \$t: a comma-delimited string hinting at hits per module OR empty

What needs to be done is to query the relevant search function from one or more modules and present them to the visitor via \$theme.

Note: array \$qwords contains 1 or more arrays with the original keyword (in \$qwords[i][0]), the lowercase version (in \$qwords[i][1]) and also the quoted version ready for 'LIKE' in a where-clause (in \$qwords[i][2]). This may come in handy when preparing the search context later on.

The original query is also available, in \$q. This is used to construct Previous and Next links below the search results overview.

The user requests \$r results. However, we try to find \$r+1 results. If that succeeds, we know that after the current \$r results there will be at least 1 more result (on the next page). Also this further fills the cache with hints (in parameter \$t) for the next iteration.

FALSE function search_areas(&\$hits, &\$results, \$qwords, \$limit, \$offset, \$config) [line 750]

Function Parameters:

- array **&\$hits** collects the requested search results if any
- int **&\$results** collects the # of hits in this module
- array **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- int **\$limit** maximum number of hits to return
- int **\$offset** starting point within this module for returning results
- array **\$config** search configuration data (contains scope and area_id)

search through area titles

FALSE function search_construct_nodelist(\$config) [line 593]

Function Parameters:

- array **\$config** search module configuration data

construct a helper table with a list of node_id's that are to be searched

prepare a temporary table 'search_nodes' with all the nodes that are to be searched. Nodes that are NOT to be searched are those under embargo, expired or those in areas where

we are not allowed to look.

This is quite a complex routine, mainly because MySQL won't let me use the same temporary table twice in the same SQL statement. The work-around is to have more than one temporary table to build a list of searchable nodes.

The strategy is as follows.

1. Determine the areas selected to be searched based on \$config

The search module is configured to either - search the current (single) area (config['scope'] = 0)

- search all public areas (config['scope'] = 1)
- search all areas where the current user has access (config['scope'] = 2)

Note that if the current user is not logged in, the latter two yield the same result. The area selection is kept in \$where which looks like - ((a.is_active) AND (a.area_id=AA))

- ((a.is_active) AND (a.is_private=FALSE))
- ((a.is_active) AND ((a.is_private=FALSE) OR (a.area_id=A) OR (area_id=B) OR ...))

2. Select all embargoed/expired nodes in the selected areas (into new temp table '\$accu')

This gives a list of otherwise valid nodes that happen to be embargoed/expired and therefore are not to be searched. If this list is empty we can proceed with step 6 below. Note that the temp table '\$accu' is always created, even if it is empty.

3. Select all the children from the nodes selected in the previous step into new temp table '\$work'

The effect of embargo/expiry effects all descendants of the embargoed/expired nodes. We have to work our way through all offspring of the embargoed/expired nodes to find all the nodes that are not to be searched. If the list of this generation of descendants is empty we can proceed with step 6 below.

4. Add the generation in temp table 2 ('\$work') to the family in temp table 1 ('\$accu')

This is the workaround with two temp tables. The new generation in \$work is added to the list of ancestors in \$accu and \$work is cleared for the next round.

5. Select the next generation in \$work

If there is a next generation it is stored in the existing table \$work and we continue with step 4 (again). If there are no new generations at this time, we fall through to step 6.

This loop 4+5 eventually leaves all embargoed/expired nodes and all their descendants in temp table 1 '\$accu'. Table temp table '\$work' is no longer required at that point.

6. Invert the selection of nodes currently in temp table 1 '\$accu' taking areas into account

This selects all the node_id's in the selected areas (see step 1) that are NOT in temp table 1 '\$accu' into temp table 3 ('\$nodes').

7. Return the number of records in temp_table3 '\$nodes' as the # of searchable nodes.

array function search_context(\$qwords, \$content, [\$window = 100], [\$maxruns = 5], [\$maxhits = 10]) [line 947]

Function Parameters:

- **array \$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- **string \$content** original content to search
- **int \$window** arbitrary base for size of context snippets
- **int \$maxruns** arbitrary limit for number of runs to return
- **int \$maxhits** arbitrary limit for # of hits/qword

return a ready-to-use context with highlights

search \$content for keywords in \$qwords and return highlighted snippets (if any).

Strategy is as follows:

1. prepare the \$content: make lowercase, strip HTML, etc.
if the string is now shorter than the \$window we're done:
return the highlighted string.
2. iterate through \$words and the string, rember the offset and length of any matches in array hits[]. If there are no hits we're done: return an empty array.
3. step through \$hits and attempt to combine \$hits within a \$window into a single run of hits in \$runs[] (storing starting and ending offsets).
4. step through \$runs and extract the specified substrings from the string, adding about half a window context before and about half a window context after the run. Store the highlighted strings in an array and return that array.

- Usedby [htmlpage_search\(\)](#)

bool|array function search_get_places(\$hints) [*line 474*]

Function Parameters:

- *string* **\$hints** comma delimited list of hits per module

construct an ordered list of places to search perhaps with hints about # of hits

the list of places to search starts with areas and nodes: these are special cases and these places are searched first. After that we give priority to modules htmlpage, newsletter and althing. After that the order is chosen so that the 'dynamic' pages (periodic newsletter, active weblog) are searched first. All current modules (April 2016) are added to the list in a particular order. Additional modules will be added automatically at the end of the list. This could differ between W@S installations, but the order is the same between searches in the same installation of W@S. We do set results to 0 for all modules; when a module of that name exists the results will be set to -1 indicating 'unknown'. This way a non-existing module will not yield errors.

The parameter \$hints contains the search results from a previous search. This is basically a comma delimited list of integers where the first integer is the # of hits for the areas search, the second one for the nodes search, the third one for the search in htmlpage, etc. This saves trips to the database if the user moves forward in the results list. We do have to trust the user submitted data though. OTOH: if the numbers in \$hints are too high, the search will simply fail and no harm is done.

Note: 1-on-1 means: one module record is linked to a node, 1-on-N means that more than 1 results could emanate from the same node (eg. different issues of the newsletter, or different posts in the althing). NOP indicates that that module always performs No OPeration: always 0 results and never any hits.

array function search_get_qwords(\$q) [*line 725*]

Function Parameters:

- *string* **\$q** holds a list of words to search for

construct an array with keywords to look for

construct an array with 0, 1 or more keywords to look for in the database we prepare different versions: the original casing, the lowercase variant and a version prepared for the LIKE-operator. The second comes in handy when we are highlighting the context in the results list. The third one is handy when performing database queries.

Note: we KISS and do not take quoted strings into account (for now). Note that earlier we already stripped HTML-tags from \$q. (see [search_view_dialog_validate\(\)](#)).

- **TODO** better parse of keywords taking quotes into account too.

string function search_highlight(\$qwords, \$snippet) [*line 1072*]

Function Parameters:

- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *string* **\$snippet** lowercase version of a context snippet

highlight words in a lowercase search context snippet

highlight keywords in \$snippet by sandwiching them between " **and** " we first create a list of (unique) words ordered by the lenght and subsequently use that list for matching. This list is based on the utf8lower variant of the keyword.

- **TODO** what to do with overlapping qwords, eg. "foo" and "foobar"? result depends on search/replace execution order: either "**foobar**" or "**foobar**"
- **TODO** is there a chance that our static cache \$q fails if somehow a different \$qwords is used on subsequent calls? Mmmmm...

bool|array function search_module(&\$hits, &\$results, \$qwords, \$limit, \$offset, \$module) [*line 418*]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results
- *array* **\$module** holds module information (name, search_script, module_id, etc.)

kickstarter for calling modules' search function (after loading the source files)

The interface with the modules' search functions is fully documented in [htmlpage_search\(\)](#) in [htmlpage_search.php](#).

array function search_navbar(\$count, \$r, \$o, \$params) [line 356]

Function Parameters:

- *int* **\$count** total number of hits counted sofar (could be grand total)
- *int* **\$r** results per page
- *int* **\$o** offset
- *array* **\$params** basic parameters for the URLs to generate

construct a navigation bar with 'previous' 1 2 3 4 ... 'next'

construct a DIV with 'previous' 1 2 3 4 ... 'next' links to navigate the search results. If there is less than 1 page worth of results no navigation bar is necessary (an empty array is returned). If we're on the first page, the 'previous' link is text (not a link), if we are on the last page the 'next' link is text (not a link).

The page numbers 1 2 3 4 ... form a sliding window within all pages; width is determined by the global pagination setting (default 7).

FALSE function search_nodes(&\$hits, &\$results, \$qwords, \$limit, \$offset, \$config) [line 842]

Function Parameters:

- *array* **&\$hits** collects the requested search results if any
- *int* **&\$results** collects the # of hits in this module
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return
- *int* **\$offset** starting point within this module for returning results
- *array* **\$config** search configuration data (contains scope and area_id)

search through node titles and link_texts

there are a few possibilities 1 (\$results < 0) && (\$limit <= 0): calc \$results, return
 2 (\$results < 0) && (\$limit > 0): calc \$results, retrieve \$hits, return
 3 (\$results == 0) && (\$limit <= 0): return
 4 (\$results == 0) && (\$limit > 0): return
 5 (\$results > 0) && (\$limit <= 0): return
 6 (\$results > 0) && (\$limit > 0): retrieve selected \$hits, return

Cases 3, 4 and 5 are easy: we simply trust the caller's cached \$results and we do not have to return any \$hits.

Case 1 can be done via counting the number of matches, but not retrieving them, e.g. sql ==
 SELECT count(node_id) AS n FROM \$table WHERE field LIKE '%expression%';

Case 2 is a little more difficult because we need to retrieve the records AND calculate the total number of hits. The latter is the # of records in the result set and \$limit could be (substantially) less than that.

Case 6 is a matter of selecting a \$limit hits, starting at offset. No calculations necessary and thus relatively easy.

In order to keep this code simple we combine cases 1 and 2 (partly) and cases 6 and 2. In other words: if necessary we calculate the # of records (case 1, case 2) if necessary we snatch \$limit records at \$offset (case 2, case 6) This is two trips to the database in case 2, but only on the initial search. (the subsequent runs use the cached # of hits via the \$hints)

void function search_show_form(&\$theme, \$config, \$dialogdef) [*line 197*]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *array* **\$config** search configuration data
- *array* **\$dialogdef** array that defines the input fields

display the search form

bool/array function search_simple(&\$hits, &\$results, \$qwords, \$limit, \$offset, \$table, \$fields, \$where, \$order, \$node, \$ttl) [*line 1107*]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)

- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results
- *string* **\$table** module table linked 1 : 1 to nodes
- *array* **\$fields** data to retrieve, key="\$fieldname", value="m.\$fieldname"
- *string* **\$where** full where-clause without the word WHERE
- *string* **\$order** sort order
- *string* **\$node** name of field to link to nodes.node_id 1:1
- *string* **\$ttl** name of field to use as search result title

perform a simple linear search in a module table

link tables search_nodes s and nodes n to \$table m, using \$node as the link field. retrieves hits in \$hits, using either "m.\$ttl" or "n.title" as title. keys in \$fields are plain fieldnames, values are prefixed with 'm.' preventing \$DB ambiguities

bool function search_view(&\$theme, \$area_id, \$node_id, \$module) [*line 47*]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

display the content of the search linked to node \$node_id

- **TODO** should we use the token routines here too?

bool function search_view_dialog_validate(&\$dialogdef) [*line 166*]

Function Parameters:

- *object* **&\$dialogdef** defines the dialog and will hold POSTed values

validate the data entered by the visitor

bool/array function `search_view_get_config($node_id, $area_id)` [line 121]

Function Parameters:

- *int* **\$node_id** identifies the page
- *int* **\$area_id** identifies the current area

retrieve all configuration data for this search

this retrieves all configuration data for this search. Here is a quick reminder of the structure in `$config`. 'node_id'

'header'

'introduction'

'scope'

'results'

As a bonus we also add the current area `$area_id` to the resulting array.

array function `search_view_get_dialogdef()` [line 143]

construct a dialog definition for the visitor's search form

this defines the searchform, basically a single text input and a [Search] button.

string function `search_where($qwords, $fields)` [line 902]

Function Parameters:

- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *array* **\$fields** holds 1 or more fieldnames

construct a where-clause to look for keywords in 1 or more fields

Possible outcomes for fields `f,f1,f2` and `qwords q,q1,q2`:

- (f LIKE %q%)
- ((f LIKE %q1%) AND (f LIKE %q2%))

- ((f1 LIKE %q%) OR (f2 LIKE %q%))
- (((f1 LIKE %q1%) AND (f1 LIKE %q2%)) OR ((f2 LIKE %q1%) AND (f2 LIKE %q2%)))

- **Usedby** [htmlpage_search\(\)](#)

Package wasmod_sitemap Procedural Elements

sitemap_tabledefs.php

/program/modules/sitemap/install/sitemap_tabledefs.php - data definition for module

- **Package** wasmod_sitemap
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: sitemap_tabledefs.php,v 1.7 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/en/sitemap.php - translated messages for module (English)

- **Package** wasmod_sitemap
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: sitemap.php,v 1.9 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap.php

/program/modules/sitemap/languages/nl/sitemap.php - translated messages for module (Dutch)

- **Package** wasmod_sitemap
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: sitemap.php,v 1.7 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap_admin.php

/program/modules/sitemap/sitemap_admin.php - management interface for sitemap-module

This file defines the administrative interface to this module. The interface consists of the following four functions.

```
sitemap_disconnect(&$output,$area_id,$node_id,$module)
sitemap_connect(&$output,$area_id,$node_id,$module)
sitemap_show_edit(&$output,$area_id,$node_id,$module,$viewonly,$edit_again,$href,$option)
sitemap_save(&$output,$area_id,$node_id,$module,$viewonly,&$edit_again,$option)
```

These functions are called from pagemanagerlib.php whenever necessary.

- **Package** wasmod_sitemap
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: sitemap_admin.php,v 1.9 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function sitemap_connect(&\$output, \$area_id, \$node_id, \$module) [*line 73*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which we need to connect
- *array* **\$module** the module record straight from the database

connect this module to a node

this makes the link between the node \$node_id in area \$area_id and this module. In this case we simply link a single 'scope' parameter to node \$node_id in a 1-to-1 relation.

Note that we set the parameter 'scope' to 0. This implies a 'small' map. It is up to the user to configure the node to use medium (scope=1) or large (scope=2) map.

bool function sitemap_disconnect(&\$output, \$area_id, \$node_id, \$module) [*line 51*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node from which we need to disconnect
- *array* **\$module** the module record straight from the database

disconnect this module from a node

this breaks the link between the node \$node_id in area \$area_id and this module. For now we simply delete the record with the sitemap scope.

array function sitemap_get_dialogdef(\$viewonly) [*line 239*]

Function Parameters:

- *int* **\$viewonly** if TRUE the Save button is not displayed and values cannot be changed by the user

construct a dialog definition for the sitemap 'scope' value

bool function sitemap_save(&\$output, \$area_id, \$node_id, \$module, \$viewonly, &\$edit_again, \$option) [*line 190*]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

save the modified content data of this module linked to node \$node_id

this validates and saves the data that was submitted by the user. If validation fails, or storing the data doesn't work, the flag \$edit_again is set to TRUE and the return value is FALSE.

If the user has cancelled the operation, the flag \$edit_again is set to FALSE and the return value is also FALSE.

If the modified data is stored successfully, the return value is TRUE and the value of \$edit_again is based on the button used to save ([Save] or [Done]).

Here is a summary of return values.

\$retval	\$edit_again	Action
-----	-----	-----
FALSE	FALSE	[Cancel] Stop editing, unlock & return to tree view
FALSE	TRUE	[Save] or [Done] failed: redo the dialog and let user correct mistakes
TRUE	FALSE	[Done] pressed, everything saved succesfully, we're done
TRUE	TRUE	[Save] pressed, everything saved succesfully but continue editing

The parameter \$option is not used in this module.

bool function sitemap_show_edit(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href, \$option)
[line 126]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

present the user with a dialog to modify the sitemap that is connected to node \$node_id

this prepares a dialog for the user filled with existing data (if any), possibly allowing the user to modify the content. If the flag \$viewonly is TRUE, this routine should only display the

content rather than let the user edit it. If the flag \$edit_again is TRUE, the routine should use the data available in the \$_POST array, otherwise it should read the data from the database (or wherever the data comes from). The parameter \$href is the place where the form should be POST'ed.

The dialog should be added to the \$output object. Useful routines are:

```
$output->add_content($content): add $content to the content area  
$output->add_message($message): add $message to the message area (feedback to the user)  
$output->add_popup_bottom($message): popup $message in browser after loading the page (uses javascript)  
$output->add_popup_top($message): popup $message in browser before loading the page (uses javascript)
```

The parameter \$option is not used in this module.

sitemap_cron.php

/program/modules/sitemap/sitemap_cron.php - interface to the cron-part of the sitemap module

This file defines the interface with the sitemap-module for cron. The interface consists of this function:

```
sitemap_cron()
```

This function is called whenever cron determines that it is time to perform this function.

- **Package** wasmod_sitemap
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: sitemap_cron.php,v 1.6 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function sitemap_cron() [*line 44*]

routine that is called periodically by cron

there is nothing in this module that needs to be done by cron, so this routine is basically a nop.

sitemap_install.php

/program/modules/sitemap/sitemap_install.php - installer of the sitemap module

This file contains the sitemap module installer. The interface consists of these functions:

```
sitemap_install(&$messages,$module_id)
sitemap_upgrade(&$messages,$module_id)
sitemap_uninstall(&$messages,$module_id)
sitemap_demodata(&$messages,$module_id,&$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wasmod_sitemap
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: sitemap_install.php,v 1.6 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function sitemap_demodata(&\$messages, \$module_id, &\$config, \$manifest) [*line 151*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table
- *array* **&\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this module

add demonstration data to the system

this routine is a no-op because all sitemap demodata is already created in the main demodata-routine in `/program/install/demodata.php`. This routine is retained here as an example alias placeholder.

This routine is retained here as an example and also because a routine by this name should exist (even if it does nothing).

Note If the module is installed via the Install Wizard, this routine is called. However, if a module is installed as an additional module after installation, the `{ $mode }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

bool function sitemap_install(&\$messages, \$module_id) [line 63]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

install the module

this routine installs the module. For this module there is nothing to install, so we simply return success. The appropriate table is already created based on the tabledefs); see `install/sitemap_tabledefs.php`.

Note that the record for this module is already created in the modules table; the pkey is `$module_id`.

bool function sitemap_uninstall(&\$messages, \$module_id) [line 121]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

uninstall the module

this is a hook for future extensions of Website@School. For now we simply return success. Any real code could look like this: `DELETE FROM sitemap;` to delete all existing data (but who would want that). For now this is simply a nop.

Note that bluntly deleting from the sitemap table might lead to nodes without a valid module.

The better way to do it would be something like this:

```
SELECT count(node_id) AS number_of_nodes FROM sitemap;
```

```
if ($number_of_nodes > 0) then
    $messages[] = 'There are still $number_of_nodes nodes with a sitemap';
    return FALSE;
```

which in fact means that the table should already be empty before we can empty it. Oh well...

bool function sitemap_upgrade(&\$messages, \$module_id) [line 91]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

upgrade the module

this routine performs an upgrade to the installed module. Note that this minimalistic 'sitemap' module does not need any upgrade at all because there never was an earlier version.

However, if there was to be a newer version of this module, this routine is THE place to bring the database up to date compared with the existing version. For example, if an additional field 'sitemap_extension' was to be added to the sitemap-table, it could be added using a suitable (default) value, e.g. an empty string or whatever

Any existing sitemap could then be updated here to fill the new field with data, e.g.

```
UPDATE sitemap SET sitemap_extension = "";
```

etcetera. For now this routine is a nop.

sitemap_manifest.php

/program/modules/sitemap/sitemap_manifest.php - description of the sitemap module

This file defines the various components of the sitemap module such as the names of the various scripts and version information. This file is used when this module is installed.

- **Package** wasmod_sitemap
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: sitemap_manifest.php,v 1.13 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

sitemap_search.php

/program/modules/sitemap/sitemap_search.php - interface to the search-part of this module

This file defines the interface with this module for searching content. The interface consists of a (temporary) table and this function:

```
bool sitemap_search(&$hits,&$results,$qwords, $limit, $offset)
```

The function is called whenever module content is to be searched. At that point the helper table, which is called 'search_nodes', exists and is populated.

- **Package** wasmod_sitemap
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: sitemap_search.php,v 1.7 2016/04/05 14:33:01 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool|array function sitemap_search(&\$hits, &\$results, \$qwords, \$limit, [\$offset = 0]) [*line 49*]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results

search this module's content in selected nodes for keywords in \$qwords

This interface is documented in [htmlpage_search\(\)](#) in [htmlpage_search.php](#).

sitemap_view.php

/program/modules/sitemap/sitemap_view.php - interface to the view-part of the sitemap module

This file defines the interface with the sitemap-module for viewing content. The interface consists of this function:

```
sitemap_view(&$output,$area_id,$node_id,$module)
```

This function is called from /index.php when the node to display is connected to this module.

- **Package** wasmod_sitemap
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: sitemap_view.php,v 1.7 2016/03/23 09:28:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

void function sitemap_tree_walk(&\$theme, \$subtree_id, &\$tree, [\$m = ""]) [line 155]

Function Parameters:

- *object* **&\$theme** collects the output
- *int* **\$subtree_id** where to start walking the the tree
- *array* **&\$tree** the structure that holds the tree
- *string* **\$m** improves readability in output

walk the tree and send to output in the form of nested unnumbered lists (uses recursion)

- **Usedby** [sitemap_tree_walk\(\)](#)
- **Uses** [sitemap_tree_walk\(\)](#)

bool function sitemap_view(&\$theme, \$area_id, \$node_id, \$module) [line 54]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

display the content of the sitemap linked to node \$node_id

there are three different variations (depends on configuration parameter 'scope'):

- 0 (small): only show a map of the tree in the current area \$area_id
- 1 (medium): show a list of available areas followed by the map of the current area \$area_id
- 2 (large): show the maps of all available areas
The default is 0 (small).

Package wasmod_snapshots Procedural Elements

snapshots_tabledefs.php

/program/modules/snapshots/install/snapshots_tabledefs.php - data definition for module

- **Package** wasmod_snapshots
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: snapshots_tabledefs.php,v 1.4 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots.php

/program/modules/snapshots/languages/en/snapshots.php - translated messages for module (English)

- **Package** wasmod_snapshots
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: snapshots.php,v 1.9 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots.php

/program/modules/snapshots/languages/nl/snapshots.php - translated messages for module (Dutch)

- **Package** wasmod_snapshots
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: snapshots.php,v 1.7 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots_admin.php

/program/modules/snapshots/snapshots_admin.php - management interface for snapshots-module

This file defines the administrative interface to this module. The interface consists of the following four functions.

```
snapshots_disconnect(&$output,$area_id,$node_id,$module)
snapshots_connect(&$output,$area_id,$node_id,$module)
snapshots_show_edit(&$output,$area_id,$node_id,$module,$viewonly,$edit_again,$href,$option)
snapshots_save(&$output,$area_id,$node_id,$module,$viewonly,&$edit_again,$option)
```

These functions are called from pagemanagerlib.php whenever necessary.

- **Package** wasmod_snapshots
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: snapshots_admin.php,v 1.6 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function snapshots_check_path(&\$item, \$area_id, \$node_id) [line 389]

Function Parameters:

- *array* **&\$item** holds the field definition from the \$dialogdef for the snapshots_path
- *int* **\$area_id** the area in which we are editing a snapshot module configuration
- *int* **\$node_id** the node to which the snapshot module is connected (unused)

validate and massage the user-supplied data path

this checks the directory path the user entered, returns TRUE if the tests are passed.

There three places from which snapshots can be retrieved:

- /areas/aaa
- /users/uuu
- /groups/ggg

That is: the path should at least contain 2 levels (and possibly more). In other words: a bare '/' is not enough and neither are bare '/areas', '/users' or '/groups'. And of course the

directory should already exist in the file system under \$CFG->datadir.

Various tests are done: - the selected area directory must be active - if the selected area is private, \$USER must have intranet access for this area, OR the selected area must be the same as the area in which \$node_id resides - the selected user directory must be the \$USER's, OR the \$USER has access to the account manager (able to manipulate ALL users' directories) - the selected group directory must be from a group the \$USER is a member of, OR the \$USER has access to the account manager (able to manipulate ALL groups' directories)

If all tests succeed, we may want to warn the user in the case that the file location is in a different (and public) area than the node holding the snapshots module. However, this is a warning only.

Finally, we reconstruct the path in such a way that it starts with a slash and does NOT end with a slash. This is done by changing the content of the \$item parameter.

- **TODO** should the user / group really be active here? If not, the images will fail in file.php but that may leak information about inactive users. Hmmm...
- **TODO** we should use a different error message as soon as it is available in was.php, eg. 'validate_bad_directory' (much like 'validate_bad_filename').

bool function snapshots_connect(&\$output, \$area_id, \$node_id, \$module) [line 84]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which we need to connect
- *array* **\$module** the module record straight from the database

connect this module to a node

this makes the link between the node \$node_id in area \$area_id and this module. In this case we simply link a single 'variant' parameter to node \$node_id in a 1-to-1 relation.

Note that we set the parameter 'variant' to 1. This equates to the variant where the visitor starts with the title, the optional introductory text and the thumbnail overview. It is up to the user to configure the node to use other variants, eg. start at the first picture full-size or display

a slide show. Also note that we start off with an (arbitrary) dimension for the full-size snapshots. This is a per-node setting (as opposed to the systemwide setting for thumbnail dimensions). Finally, we do a little heuristics here by plugging in the current directory from the filemanager. This is dirty, but we might assume that the user uploaded the files to a directory just before adding this snapshots node. In that case there is no need to change anything re: path. If the user did NOT upload files, we plug in the name of the directory which is associated with area \$area_id.

bool function snapshots_disconnect(&\$output, \$area_id, \$node_id, \$module) [line 51]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node from which we need to disconnect
- *array* **\$module** the module record straight from the database

disconnect this module from a node

this breaks the link between the node \$node_id in area \$area_id and this module. For now we simply delete the record with the snapshots variant + introduction.

array function snapshots_get_dialogdef(\$viewonly) [line 274]

Function Parameters:

- *int* **\$viewonly** if TRUE the Save button is not displayed and config values cannot be changed

construct a dialog definition for the snapshots configuration data

bool function snapshots_save(&\$output, \$area_id, \$node_id, \$module, \$viewonly, &\$edit_again, \$option) [line 213]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which the content is connected
- *array* **\$module** the module record straight from the database

- *bool* **\$viewonly** if TRUE, editing and hence saving is not allowed
- *bool* **&\$edit_again** set to TRUE if we need to edit the content again, FALSE otherwise
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

save the modified content data of this module linked to node \$node_id

this validates and saves the data that was submitted by the user. If validation fails, or storing the data doesn't work, the flag \$edit_again is set to TRUE and the return value is FALSE.

If the user has cancelled the operation, the flag \$edit_again is set to FALSE and the return value is also FALSE.

If the modified data is stored successfully, the return value is TRUE and the value of \$edit_again is based on the button used to save ([Save] or [Done]).

Here is a summary of return values.

\$retval	\$edit_again	Action
-----	-----	-----
FALSE	FALSE	[Cancel] Stop editing, unlock & return to tree view
FALSE	TRUE	[Save] or [Done] failed: redo the dialog and let user correct mistakes
TRUE	FALSE	[Done] pressed, everything saved succesfully, we're done
TRUE	TRUE	[Save] pressed, everything saved succesfully but continue editing

The parameter \$option is not used in this module.

bool function snapshots_show_edit(&\$output, \$area_id, \$node_id, \$module, \$viewonly, \$edit_again, \$href, \$option) [line 146]

Function Parameters:

- *object* **&\$output** collects the html output (if any)
- *int* **\$area_id** the area in which \$node_id resides
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *bool* **\$viewonly** if TRUE, editing is not allowed (but simply showing the content is allowed)
- *bool* **\$edit_again** if TRUE start with data from \$_POST, else use data from database
- *string* **\$href** the action property of the HTML-form, the place where data will be POST'ed
- *string* **\$option** indicates which submenu-option was selected (NULL for none)

present the user with a dialog to modify the snapshots that are connected to node \$node_id

this prepares a dialog for the user filled with existing data (if any), possibly allowing the user to modify the content. If the flag \$viewonly is TRUE, this routine should only display the content rather than let the user edit it. If the flag \$edit_again is TRUE, the routine should use the data available in the \$_POST array, otherwise it should read the data from the database (or wherever the data comes from). The parameter \$href is the place where the form should be POST'ed.

The dialog should be added to the \$output object. Useful routines are:

```
$output->add_content($content): add $content to the content area  
$output->add_message($message): add $message to the message area (feedback to the user)  
$output->add_popup_bottom($message): popup $message in browser after loading the page (uses javascript)  
$output->add_popup_top($message): popup $message in browser before loading the page (uses javascript)
```

The parameter \$option is not used in this module.

- **TODO** we might want to jazz up this dialog by adding some sort of 'directory browser' for the snapshots_path field using a pop-up window. Mmmmm.... future refinements...

snapshots_cron.php

/program/modules/snapshots/snapshots_cron.php - interface to the cron-part of the snapshots module

This file defines the interface with the snapshots-module for cron. The interface consists of this function:

```
snapshots_cron()
```

This function is called whenever cron determines that it is time to perform this function.

- **Package** wasmod_snapshots
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: snapshots_cron.php,v 1.5 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function snapshots_cron() [*line 44*]

routine that is called periodically by cron

there is nothing in this module that needs to be done by cron, so this routine is basically a nop.

snapshots_install.php

/program/modules/snapshots/snapshots_install.php - installer of the snapshots module

This file contains the snapshots module installer. The interface consists of these functions:

```
snapshots_install(&$messages,$module_id)
snapshots_upgrade(&$messages,$module_id)
snapshots_uninstall(&$messages,$module_id)
snapshots_demodata(&$messages,$module_id,&$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wasmod_snapshots
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: snapshots_install.php,v 1.7 2016/06/16 13:48:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function snapshots_demodata(&\$messages, \$module_id, &\$config, \$manifest) [line 184]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table
- *array* **&\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy of the manifest for this module

add demonstration data to the system

this routine adds to the existing set of demonstration data as specified in `$config`. Here we add a node as follows:

'snapshots': a page in section 'showcase' acting as an example set of snapshots

The page 'snapshots' is added at the bottom of the demo-section 'showcase' in the MyPage section. This is our (new) place to show off new modules.

Note If the module is installed via the Install Wizard, this routine is called. However, if a module is installed as an additional module after installation, the `{ $module }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

The array `$config` contains the following information.

```
$config['language_key'] => install language code (eg. 'en')
$config['dir']           => path to CMS Root Directory (eg. /home/httpd/htdocs)
$config['www']           => URL of CMS Root Directory (eg. http://exemplum.eu)
$config['progdir']       => path to program directory (eg. /home/httpd/htdocs/program)
$config['progwww']       => URL of program directory (eg. http://exemplum.eu/program)
$config['datadir']       => path to data directory (eg. /home/httpd/wasdata/alb2c3d4e5f6)
$config['user_username'] => userid of webmaster (eg. wblader)
$config['user_full_name'] => full name of webmaster (eg. Wilhelmina Bladergroen)
$config['user_email']    => email of webmaster (eg. w.bladergroen@exemplum.eu)
$config['user_id']       => numerical user_id (usually 1)
$config['demo_salt']     => password salt for all demodata accounts
$config['demo_password'] => password for all demodata accounts
$config['demo_areas']    => array with demo area data
$config['demo_groups']   => array with demo group data
$config['demo_users']    => array with demo user data
$config['demo_nodes']    => array with demo node data
$config['demo_string']   => array with demo strings from /program/install/languages/LL/demodata.php
$config['demo_replace']  => array with search/replace pairs to 'jazz up' the demo strings
```

With this information, we can add a demonstration configuration for the public area, which shows off the possibilities. Note that we add our own additions to the array `&$config` so other modules and themes can determine the correct status quo w.r.t. the demodata nodes etc.

Update June 2016: we now add a second demo page, in the main tree under 'News' - 'Archives' called 'snapshots2'. This is basically a copy of the first snapshots demo but showing a different variant (just for fun).

bool function `snapshots_install(&$messages, $module_id)` [line 63]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$module_id** the key for this module in the modules table

install the module

this routine installs the module. For this module there is nothing to install, so we simply return success. The appropriate table is already created based on the tabledefs); see `install/snapshots_tabledefs.php`.

Note that the record for this module is already created in the modules table; the pkey is \$module_id.

bool function snapshots_uninstall(&\$messages, \$module_id) [line 121]

Function Parameters:

- *array &\$messages* collects the (error) messages
- *int \$module_id* the key for this module in the modules table

uninstall the module

this is a hook for future extensions of Website@School. For now we simply return success. Any real code could look like this: DELETE FROM snapshots; to delete all existing data (but who would want that). For now this is simply a nop.

Note that bluntly deleting from the snapshots table might lead to nodes without a valid module. The better way to do it would be something like this:

```
SELECT count(node_id) AS number_of_nodes FROM snapshots;
```

```
if ($number_of_nodes > 0) then
    $messages[] = 'There are still $number_of_nodes nodes with a snapshots';
return FALSE;
```

which in fact means that the table should already be empty before we can empty it. Oh well...

bool function snapshots_upgrade(&\$messages, \$module_id) [line 91]

Function Parameters:

- *array &\$messages* collects the (error) messages
- *int \$module_id* the key for this module in the modules table

upgrade the module

this routine performs an upgrade to the installed module. Note that this minimalistic 'snapshots' module does not need any upgrade at all because there never was an earlier version.

However, if there was to be a newer version of this module, this routine is THE place to bring the database up to date compared with the existing version. For example, if an additional field 'snapshots_extension' was to be added to the snapshots-table, it could be added using a suitable (default) value, e.g. an empty string or whatever

Any existing snapshots could then be updated here to fill the new field with data, e.g.

```
UPDATE snapshots SET snapshots_extension = ";
```

etcetera. For now this routine is a nop.

snapshots_manifest.php

/program/modules/snapshots/snapshots_manifest.php - description of the snapshots module

This file defines the various components of the snapshots module such as the names of the various scripts and version information. This file is used when this module is installed.

- **Package** wasmod_snapshots
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: snapshots_manifest.php,v 1.8 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

snapshots_search.php

/program/modules/snapshots/snapshots_search.php - interface to the search-part of this module

This file defines the interface with this module for searching content. The interface consists of a (temporary) table and this function:

```
bool snapshots_search(&$hits,&$results,$qwords, $limit, $offset)
```

The function is called whenever module content is to be searched. At that point the helper table, which is called 'search_nodes', exists and is populated.

- **Package** wasmod_snapshots
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: snapshots_search.php,v 1.6 2016/04/05 14:33:01 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool|array function snapshots_search(&\$hits, &\$results, \$qwords, \$limit, [\$offset = 0]) [line 49]

Function Parameters:

- *array* **&\$hits** receives search results (at most \$limit)
- *int* **&\$results** receives number of hits (negative value on entry means (re-)calculate total)
- *array* **\$qwords** holds 1 or more keywords to search for (original, utf8lower, quoted)
- *int* **\$limit** maximum number of hits to return (could be 0)
- *int* **\$offset** starting point within this module for returning results

search this module's content in selected nodes for keywords in \$qwords

This interface is documented in [htmlpage_search\(\)](#) in [htmlpage_search.php](#).

snapshots_view.php

/program/modules/snapshots/snapshots_view.php - interface to the view-part of the snapshots module

This file defines the interface with the snapshots-module for viewing content. The interface consists of this function:

```
snapshots_view(&$output,$area_id,$node_id,$module)
```

This function is called from /index.php when the node to display is connected to this module. Internally all the work is done in the Snapshot class. This class is also used in the module that aggregates different nodes into a single HTML-document.

- **Package** wasmod_snapshots
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: snapshots_view.php,v 1.7 2016/04/08 12:06:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function snapshots_view(&\$theme, \$area_id, \$node_id, \$module) [line 62]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

display the snapshots from the directory linked to node \$node_id

this routine is only a helper to create a new SnapshotViewer instance which is where the real work is done.

in general there are three different variants (depends on configuration of 'variant'):

- 1 (thumbs): show the title, the introduction and thumbnails of all snapshots

- 2 (first): show the first snapshot from the series full-size
- 3 (slideshow): automatically rotate through all snapshots (uses javascript)
The default is 1 (thumbs).

The 4th variant is used in asynchronous updating of the list of images. Calls to this variant originate in slideshow.js.

Package wasmod_snapshots Classes

Class SnapshotViewer

[line 70]

this class implements methods to display snapshots

- **Package** wasmod_snapshots

SnapshotViewer::\$area_id

int = [line 75]

- **Var** \$area_id indicates the working area

SnapshotViewer::\$default_showtime

int = 5 [line 105]

- **Var** \$default_showtime is the default # of seconds between images in a slideshow

SnapshotViewer::\$dimension

int = [line 96]

- **Var** \$dimension defines the box size for variant 2

SnapshotViewer::\$domain

string = [line 84]

- **Var** \$domain language domain where we get our translations from, usually 'm_<modulename>'

SnapshotViewer::\$header

string = [line 87]

- **Var** \$header the (optional) title to display

SnapshotViewer::\$introduction

string = [line 90]

- **Var** \$introduction the (optional) introductory text to display

SnapshotViewer::\$module_record

array = [line 81]

- **Var** \$module_record the module record straight from the database

SnapshotViewer::\$node_id

int = [line 78]

- **Var** \$node_id indicates the node associated with the snapshots

SnapshotViewer::\$snapshots

null|array = NULL [line 102]

- **Var** \$snapshots snapshot information from snapshot directory or NULL if not yet set

SnapshotViewer::\$snapshots_path

string = [line 99]

- **Var** \$snapshots_path the directory containing the snapshots

SnapshotViewer::\$theme

\$theme = [line 72]

- **Var** collects the (html) output

SnapshotViewer::\$variant

int = [line 93]

- **Var** \$variant 1=thumbnails, 2=first, 3=slideshow

Constructor *void* function `SnapshotViewer::SnapshotViewer(&$theme, $area_id, $node_id, $module)` [line 115]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives (currently unused)
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database

the constructor only stores relevant data for future use

void function `SnapshotViewer::add_snapshot_navbar($snapshot_index, [$m = ""])` [line 349]

Function Parameters:

- *int* **\$snapshot_index** indicates which element of \$snapshots contains the current snapshot
- **\$m**

add a navigation bar / tool bar for a snapshot

this bar contains the following elements:

- double arrow left links to the first snapshot in the series
- left arrow links to the previous snapshot in the series
- up arrow links to the thumbnails overview
- right arrow links to the next snapshot in the series
- double right arrow links to the last snapshot in the series
- position indicator (snapshot i from n snapshots) in the form: i/n

- **TODO** clean up this ugly code

array function `SnapshotViewer::get_snapshots($path)` [line 475]

Function Parameters:

- *string* **\$path** the directory to scan (relative to \$CFG->datadir)

retrieve all image files (snapshots) from directory \$path

This creates an array containing a (filtered) listing of the images in the directory called \$path. These items are suppressed:

- current directory '.'
- parent directory '..'
- index.html (used to 'protect' directory against prying eyes)
- symbolic links

The files THUMBNAIL_PREFIX* (the thumbnails of images) are a special case: these are used to show a small image in the thumbnail overview.

A second filtering makes sure that the items returned are actually images.

- **Uses** \$CFG;

void function SnapshotViewer::get_snapshots_configuration(\$node_id) [line 633]

Function Parameters:

- *int* **\$node_id** this key identifies the snapshots series

retrieve configuration data for this set of snapshots

this routine fetches the configuration from the snapshots table and stores the sanitised information from the various fields in the object variables.

- **TODO** check for information leaks (private path) here?

array function SnapshotViewer::images_array() [line 594]

construct an image configuration array

this steps through the snapshots list and prepares an array. The n'th image (starting at 0) in that array is defined as follows:

img[n][0] = width of the image (in pixels) img[n][1] = height of the image (in pixels) img[n][2] = the url of the image file (src-attribute of the img tag) img[n][3] = the number of seconds to

display this image `img[n][4] = title` to add to the display (document title)

void function SnapshotViewer::javascript_add_configuration(\$imgs, [\$m = ""]) [line 558]

Function Parameters:

- *array \$imgs* contains array with values to convey to javascript code in an array `img[]`
- *string \$m* code readability

construct configuration data for javascript processing

this outputs JavaScript code that is necessary to setup a slideshow. We create a 0-based array `img[]` with information about all available images. We also plug in a few translations and our own URL-variant.

void function SnapshotViewer::javascript_include_once(\$filename) [line 676]

Function Parameters:

- *string \$filename* name of the js-file relative to /program directory

include an external javascript file once

this adds an inclusion of a javascript file once in the document we are creating in `$this->theme`. If multiple instances of this SnapshotViewer-class exist the file is included only once.

bool function SnapshotViewer::run() [line 131]

task dispatcher

this routine decides what to do and calls the appropriate workhorse routine(s)

- **TODO** check permissions (ACL) to prevent leaking a private area path to anonymous visitors?

bool function SnapshotViewer::view_raw(\$images) [line 322]

Function Parameters:

- `array $images` contains 2-D array (see [images_array\(\)](#))

output the raw image data as a tab-delimited file

output format ('|' denotes TAB-character): width | height | image url | displaytime | title

bool function SnapshotViewer::view_slideshow() [line 306]

show the regular thumbnails overview and then pop-up a full-screen slideshow on top

this is basically the same as the thumbnail overview, be it that we get the effect of 'automagically' entering the slideshow (take it from the top).

bool function SnapshotViewer::view_snapshot(\$snapshot_index) [line 240]

Function Parameters:

- *int \$snapshot_index* indicates the snapshot to show

display a single full-size snapshot scaled to the specified dimension

bool function SnapshotViewer::view_thumbnails() [line 191]

display snapshots in the form of 0 or more clickable thumbnails

Class SnapshotViewerInline

[line 693]

this class implements methods to display snapshots

- **Package** wasmod_snapshots

SnapshotViewerInline::\$inline_show_height

int = 120 [line 699]

- **Var** \$inline_show_height is the available height in the inline slideshow

SnapshotViewerInline::\$inline_show_visible_images

int = 1 [line 702]

- **Var** \$inline_show_visible_images is the number of images to show simultaneously

SnapshotViewerInline::\$inline_show_width

int = 120 [line 696]

- **Var** \$inline_show_width is the available width in the inline slideshow

Constructor *void* function SnapshotViewerInline::SnapshotViewerInline(&\$theme, \$area_id, \$node_id, \$module, \$width, \$height, \$visible) [line 715]

Function Parameters:

- *object* **&\$theme** collects the (html) output
- *int* **\$area_id** identifies the area where \$node_id lives (currently unused)
- *int* **\$node_id** the node to which this module is connected
- *array* **\$module** the module record straight from the database
- *int* **\$width** the available width for the inline slideshow
- *int* **\$height** the available height for the inline slideshow
- *int* **\$visible** the # of visible images in the inline slideshow

the constructor only stores relevant data for future use

`void function SnapshotViewerInline::javascript_add_inline_show($imgs, [$m = ""]) [line 773]`

Function Parameters:

- *array* **\$imgs** contains array with values to convey to javascript code
- *string* **\$m** code readability

construct the necessary Javascript-code to do the inline slideshow configuration

this steps through the snapshots list and prepares the necessary javascript function calls to create and populate the inline slideshow. The following slideshow parameters are conveyed:

- the available width
 - the available height
 - the number of visible images
- The following image parameters are conveyed:
- width of the image (in pixels)
 - height of the image (in pixels)
 - the url of the image file (src-attribute of the img tag)
 - the number of seconds to display this image
 - title to add to the display (document title)

`void function SnapshotViewerInline::noscript_add_inline_show($imgs, [$m = ""]) [line 822]`

Function Parameters:

- *array* **\$imgs** contains array with values for static images
- *string* **\$m** code readability

construct the necessary code to show N static images in case JavaScript is OFF

this steps through the first N images of the snapshots list and prepares the necessary code to create and populate N static non-rotating img-tags, sandwiched in a noscript tag (graceful degradation...)

The following slideshow parameters are used:

- the available width
 - the available height
 - the number N of visible images
- The following parameters of the first N images are used:
- width of the image (in pixels)
 - height of the image (in pixels)
 - the url of the image file (src-attribute of the img tag)

- title to add to the display (document title)
Notably absent:
- the number of seconds to display this image ;-)

bool function SnapshotViewerInline::run() [*line 730*]

read configuration parameters and actually generate the inline slide show
this routine decides what to do and calls the appropriate workhorse routine(s)

- **TODO** check permissions (ACL) to prevent leaking a private area path to anonymous visitors?

Package wastheme_axis Procedural Elements

axis.class.php

/program/themes/axis/axis.class.php - implements the Axis Theme

- **Package** wastheme_axis
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: axis.class.php,v 1.4 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

axis_install.php

/program/themes/axis/axis_install.php -- installer of the Axis Theme

This file contains the Axis Theme installer. The interface consists of these functions:

```
axis_install(&$messages,$theme_id)
axis_upgrade(&$messages,$theme_id)
axis_uninstall(&$messages,$theme_id)
axis_demodata(&$messages,$theme_id,$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wastheme_axis
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: axis_install.php,v 1.4 2016/03/23 09:28:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function axis_demodata(&\$messages, \$theme_id, \$config, \$manifest) [line 234]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table
- *array* **\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this theme

add demonstration data to the system

this routine adds demonstration data to the freshly installed system

Note If the theme is installed via the Install Wizard, this routine is called. However, if a theme

is installed as an additional theme after installation, the `{ $theme }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

The array `$config` contains the following information.

```
$config['language_key'] => install language code (eg. 'en')
$config['dir']          => path to CMS Root Directory (eg. /home/httpd/htdocs)
$config['www']          => URL of CMS Root Directory (eg. http://exemplum.eu)
$config['progdir']      => path to program directory (eg. /home/httpd/htdocs/program)
$config['progwww']      => URL of program directory (eg. http://exemplum.eu/program)
$config['datadir']      => path to data directory (eg. /home/httpd/wasdata/alb2c3d4e5f6)
$config['user_username'] => userid of webmaster (eg. wblader)
$config['user_full_name'] => full name of webmaster (eg. Wilhelmina Bladergroen)
$config['user_email']   => email of webmaster (eg. w.bladergroen@exemplum.eu)
$config['user_id']      => numerical user_id (usually 1)
$config['demo_salt']    => password salt for all demodata accounts
$config['demo_password'] => password for all demodata accounts
$config['demo_areas']   => array with demo area data
$config['demo_groups']  => array with demo group data
$config['demo_users']   => array with demo user data
$config['demo_nodes']   => array with demo node data
```

With this information, we can add a demonstration configuration for the public area, which shows off the possibilities.

array function `axis_get_properties()` [line 291]

construct a list of default properties for this theme

this routine is used when installing axis for the first time and also when upgrading the theme to the latest version

Note that this (simple) theme does not have 'fancy' features like quicklinks at the top or the bottom or a logo; it is really a 1 or 2 level theme with no frills.

- Used by [axis_upgrade\(\)](#)
- Used by [axis_install\(\)](#)

bool function `axis_install(&$messages, $theme_id)` [line 64]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

install the theme

this routine performs the necessary actions to make this theme usable. More specific, this routine adds a handful of default values into the themes_properties table. Once a theme is actually used in an area, these defaults are copied from the themes_properties table to the themes_areas_properties table for the selected area. The user can subsequently edit these properties in the Area Manager.

- Uses [axis_get_properties\(\)](#)

bool function axis_uninstall(&\$messages, \$theme_id) [line 185]

Function Parameters:

- *array &\$messages* collects the (error) messages
- *int \$theme_id* the key for this theme in the themes table

uninstall the theme

this is a hook for future extensions of Website@School. For now we simply return success. Any real code could look like this:

```
DELETE FROM themes_areas_properties WHERE theme_id = $theme_id; DELETE FROM
themes_properties WHERE theme_id = $theme_id; DELETE FROM themes WHERE
theme_id = $theme_id;
```

or whatever.

bool function axis_upgrade(&\$messages, \$theme_id) [line 124]

Function Parameters:

- *array &\$messages* collects the (error) messages
- *int \$theme_id* the key for this theme in the themes table

upgrade the theme

this routine performs an upgrade to the installed theme.

Note that the initial version of this 'axis' theme does not need any upgrade at all because there never was an earlier version (well, duh).

However, if there was to be a newer version of this theme, this routine is THE place to bring the database up to date compared with the existing version. For example, if an additional property 'foobar' was to be added to the theme configuration, it could be added to the themes_properties table with a suitable (default) value. Any existing areas with this theme could have their configuration updated with this additional foobar property, e.g. INSERT INTO themes_properties: foobar for all areas in themes_areas_properties with theme_id = \$theme_id do INSERT INTO themes_areas_properties: foobar etcetera,

The current version of the theme could be determined by consulting the database (db_select_single_record(themes, '*', 'theme_id = \$theme_id') etcetera.

Note that it is the responsibility of the caller to correctly store the data from the manifest in the themes table. You should not do this here, in this routine.

Currently this is a quick and dirty routine to

- update changed sort_order in the existing settings, OR
- add fields that were not available in the current settings

In the future we could make it more sophisticated by updating the themes_areas_properties too. Oh well. KISS

- **TODO** maybe make this a little less quick and dirty?
- **Uses** [axis_get_properties\(\)](#)

axis_manifest.php

/program/themes/axis/axis_manifest.php - description of the axis theme

This file defines the 'axis' theme. This file is used when this theme is installed or upgraded.

- **Package** wastheme_axis
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: axis_manifest.php,v 1.8 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php

/program/themes/axis/languages/en/axis.php - translated messages for theme (English)

- **Package** wastheme_axis
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: axis.php,v 1.6 2016/03/23 09:28:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

axis.php
/program/themes/axis/languages/nl/axis.php - translated messages for theme
(Nederlands)

- **Package** wastheme_axis
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: axis.php,v 1.5 2016/03/23 09:28:27 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wastheme_axis Classes

Class ThemeAxis

[line 30]

this class implements the axis theme

- **Package** wastheme_axis

string function ThemeAxis::axis_logout([\$separator = "], [\$m = "]) [line 173]

Function Parameters:

- *string* **\$separator** a visual separator that is appended
- *string* **\$m** margin for increased readability

conditionally construct a logout link

this link is added on the left of the 'powered by websiteatschool' and 'print' widgets at the bottom of the page, but only if the current user is logged in.

string function ThemeAxis::axis_printpage([\$separator = "], [\$m = "]) [line 145]

Function Parameters:

- *string* **\$separator** a visual separator that is prepended
- *string* **\$m** margin for increased readability

construct a 'print this page' link

this link is added on the left of the 'powered by websiteatschool' widget at the bottom of the page, unless we are already creating a print version.

string function ThemeAxis::get_html() [*line 55*]

construct an output page in HTML

This constructs a full HTML-page, starting at the DTD and ending with the html closing tag.

This routine returns a full HTML-page, including a navigation menu and a footer with links to printer friendly version + logout. If the page is called with print=1 as one of the get-parameters, the background and navigation menu are suppressed by including the additional print.css stylesheet (configurable). This more or less allows for making a clean print of only the content. This additional stylesheet is added only once, even if this routine is called more than once (shouldn't happen). This stylesheet is configurable just like the regular stylesheet.

Suppressing the background image (for printing) involves NOT generating the container div with id="page", or rather: we use a different id when viewed in regular mode (id="page") or when viewed in print mode (id="print"). This allows for different tricks in print.css and at the very least allows for suppressing the background image.

Package wastheme_cornelia Procedural Elements

cornelia_install.php

/program/themes/cornelia/cornelia_install.php -- installer for the cornelia theme

This file contains the cornelia theme installer. The interface consists of these functions:

```
cornelia_install(&$messages,$theme_id)
cornelia_upgrade(&$messages,$theme_id)
cornelia_uninstall(&$messages,$theme_id)
cornelia_demodata(&$messages,$theme_id,$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wastheme_cornelia
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: cornelia_install.php,v 1.4 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function cornelia_demodata(&\$messages, \$theme_id, \$config, \$manifest) [*line 443*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

- array **\$config** pertinent data for the new website + demodata foundation
- array **\$manifest** a copy from the manifest for this theme

add demonstration data to the system

this routine adds demonstration data to the freshly installed system

Note If the theme is installed via the Install Wizard, this routine is called. However, if a theme is installed as an additional theme after installation, the `{ $theme }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

The array `$config` contains the following information.

```
$config['language_key'] => install language code (eg. 'en')
$config['dir']           => path to CMS Root Directory (eg. /home/httpd/htdocs)
$config['www']          => URL of CMS Root Directory (eg. http://exemplum.eu)
$config['progdir']      => path to program directory (eg. /home/httpd/htdocs/program)
$config['progwww']      => URL of program directory (eg. http://exemplum.eu/program)
$config['datadir']      => path to data directory (eg. /home/httpd/wasdata/alb2c3d4e5f6)
$config['title']        => the name of the site (as entered by Wilhelmina Bladergroen)
$config['user_username'] => userid of webmaster (eg. wblader)
$config['user_full_name'] => full name of webmaster (eg. Wilhelmina Bladergroen)
$config['user_email']   => email of webmaster (eg. w.bladergroen@exemplum.eu)
$config['user_id']      => numerical user_id (usually 1)
$config['demo_salt']    => password salt for all demodata accounts
$config['demo_password'] => password for all demodata accounts
$config['demo_areas']   => array with demo area data
$config['demo_groups']  => array with demo group data
$config['demo_users']   => array with demo user data
$config['demo_nodes']   => array with demo node data
$config['demo_string']  => array with demo strings from /program/install/languages/LL/demodata.php
$config['demo_replace'] => array with search/replace pairs to 'jazz up' the demo strings
```

With this information, we can add a demonstration configuration for the public area, which shows off the possibilities.

bool function `cornelia_install(&$messages, $theme_id)` [line 61]

Function Parameters:

- array **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

install the theme

this routine performs the necessary actions to make this theme usable. More specific, this routine adds a handful of default values into the `themes_properties` table. Once a theme

is actually used in an area, these defaults are copied from the themes_properties table to the themes_areas_properties table for the selected area. The user can subsequently edit these properties in the Area Manager.

```
bool function cornelia_uninstall(&$messages, $theme_id) [line 391]
```

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

uninstall the theme

this is a hook for future extensions of Website@School. For now we simply return success. Any real code could look like this:

```
DELETE FROM themes_areas_properties WHERE theme_id = $theme_id; DELETE FROM
themes_properties WHERE theme_id = $theme_id; DELETE FROM themes WHERE
theme_id = $theme_id;
```

or whatever.

```
bool function cornelia_upgrade(&$messages, $theme_id) [line 369]
```

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

upgrade the theme

this routine performs an upgrade to the installed theme. Note that this minimalistic 'cornelia' theme does not need any upgrade at all because there never was an earlier version.

However, if there was to be a newer version of this theme, this routine is THE place to bring the database up to date compared with the existing version. For example, if an additional property 'foobar' was to be added to the theme configuration, it could be added to the themes_properties table with a suitable (default) value, Any existing areas with this theme could have their configuration updated with this additional foobar property, e.g. INSERT INTO themes_properties: foobar for all areas in themes_areas_properties with theme_id = \$theme_id do INSERT INTO themes_areas_properties: foobar etcetera,

The current version of the theme could be determined by consulting the database

(db_select_single_record(themes,'*', 'theme_id = \$theme_id') etcetera.

Note that it is the responsibility of the caller to correctly store the data from the manifest in the themes table. You don't have to do this here, in this routine.

cornelia_manifest.php

/program/themes/cornelia/cornelia_manifest.php - description of the cornelia theme

This file defines the cornelia theme. It is used when this theme is installed.

- **Package** wastheme_cornelia
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: cornelia_manifest.php,v 1.7 2016/06/28 13:47:25 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/en/cornelia.php - translated messages for theme (English)

- **Package** wastheme_cornelia
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: cornelia.php,v 1.4 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

cornelia.php

/program/themes/cornelia/languages/nl/cornelia.php - translated messages for theme (Nederlands)

- **Package** wastheme_cornelia
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: cornelia.php,v 1.4 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wastheme_cornelia Classes

Class ThemeCornelia

[line 27]

/program/themes/cornelia/cornelia.class.php - the class that implements the theme

- **Package** wastheme_cornelia
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: cornelia.class.php,v 1.4 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Constructor *void* function ThemeCornelia::ThemeCornelia(\$theme_record, \$area_id, \$node_id) [line 29]

Function Parameters:

- **\$theme_record**
- **\$area_id**
- **\$node_id**

string function ThemeCornelia::cornelia_get_background_url(\$index, [\$fully_qualified = FALSE]) [line 536]

Function Parameters:

- *int* **\$index** indicates current main navigation item or 0 for none

- **`bool $fully_qualified`** if TRUE forces the URL to contain a scheme, authority etc.

compute URL for a background image based on current time and index

this routine returns a URL to be used as a background image. The URL is different depending on the time of day and the `$index` parameter. The latter is thought to be the position of the current item in the main navigation (starting at 1) or 0 in case no item in the main navigation is current.

Configuration is done via two parameters: the path in `$this->config['header_banners_directory']` and the interval in `$this->config['header_banners_interval']`.

The interval is expressed in minutes. It means that a particular page has the same background image for the duration of the interval and another one in the next interval. The banners directory is supposed to contain banner files of the correct dimensions, e.g. 980x170.

If the specified directory starts with a '/' it is assumed to start in the data root, e.g. something like `/areas/exemplum/banners`. This will eventually lead to a URL of the form `/file.php/areas/exemplum/banners/mentha-banner.jpg` using `was_file_url()`

If the specified directory does NOT start with a '/' it is assumed to be a directory relative to the directory where `index.php` and `friends` live, unless it starts with `program/` in which case it is a static dir somewhere in the program hierarchy. This is done via `was_url()`.

- Uses [was_url\(\)](#)
- Uses [was_file_url\(\)](#)

string function ThemeCornelia::cornelia_get_sidebar(\$sidebar_nodes, [\$m = ""]) [*line 432*]

Function Parameters:

- **`array $sidebar_nodes`** contains node-module pairs to show
- **`string $m`** add to human readability

retrieve data for sidebar

this retrieves the content of the various sidebar blocks (if any) currently only the `htmlpage` module is supported.

string function ThemeCornelia::cornelia_get_sidebar_htmlpage(\$index, \$node_id, [\$m = ""]) [*line 465*]

Function Parameters:

- *in* **\$index** block number (1-based)
- *int* **\$node_id** identifies the pagedata to retrieve
- *string* **\$m** increased readability

retrieve page data for htmlpage module on a node

this retrieves data from the page \$node_id (requires knowledge of internals of htmlpage module/table) in order to show that in a box in the 3rd column. The content is wrapped in a div which has a unique ID of the form sidebar_blockN where N counts from 1 upward and a common class sidebar_htmlpage. This allows for connecting style information to a particular sidebar block in a flexible way.

int function ThemeCornelia::cornelia_navigation_index() [*line 311*]

calculate the index of the current main navigation item

this routine determines which main menu item is the current one, starting at 1. if no menu item is current, the function returns 0.

This index is used for two purposes:

- calculating an ever changing background image for the header, and
- determining which item of the comma-delimited pagenumbers to use for the sidebar.

bool/array function ThemeCornelia::cornelia_sidebar_nodes_modules(\$index) [*line 364*]

Function Parameters:

- *int* **\$index** is the current main menu item or 0 for none

compute the list of sidebar blocks for a main menu item

this constructs an ordered array of node_id's corresponding to the current main menu item. If the current main menu_item indicates no sidebar then FALSE is returned.

The basis for the node_ids to return is the contents of the parameter 'sidebar_nodelist'. This is a comma-delimited string containing node_ids, zeros or dashes.

Example: suppose sidebar_nodelist contains "5,0,0,0,-". If the first item of the

main menu is current (ie. is part of the breadcrumb trail), the node with node_id = 5 is examined. If it is a page and the module is supported, this page's node_id is returned. If, however, this is a section, all supported pages within that section are returned. For main menu items 2, 3 and 4 a '0' is specified. This means that no pages will be displayed (an empty array is returned). For main menu item 5 a dash ('-') is specified. This yields a return value of FALSE which must be interpreted as 'suppress the complete 3rd column of the page, including rightmargin_top and rightmargin_bottom. This effectively reduces the 3-column layout to a 2-column layout.

Note that the node_ids in the sidebar_nodelist themselves must not be under embargo but a hidden section (with visible pages) is OK.

Note that when there are more main menu items than entries in the sidebar_nodelist, the value '0' is assumed.

Finally, if no main menu item is current (eg. the visitor arrived on a page via quicktop entries) then the value of '0' is assumed, too.

string function ThemeCornelia::get_bottomline([\$m = ""]) [line 289]

Function Parameters:

- *string* **\$m** left margin for increased readability

show footer text, maybe some quicklinks and 'powered by'

string function ThemeCornelia::get_html() [line 80]

construct an output page in HTML

This constructs a full HTML-page, starting at the DTD and ending with the html closing tag.

The page is constructed using nested DIVs, the layout is taken care of in separate style sheets. All knowledge about the structure of the page is contained in this routine.

As a rule the layout is based on three columns. The left hand column contains free form HTML, the menu and more free form HTML. The right hand column contains free form HTML, the contents of 0, 1 or more html-pages and more free form HTML. The column in the middle holds the actual content.

The list of html-pages to show in the right hand column is configurable per main menu item. A '0' (zero) means: no page, a page number means that page, a section number means all html-pages in that section and a dash ('-') means: suppress the right hand side column completely. This also inserts an additional stylesheet to style the 2-column layout. There is also a special print stylesheet. This is included after the user clicks the special print link. Note that the stylesheets are added only once, in the order 'style.css', 'style2.css' and finally 'print.css' (if

applicable).

The contents of the various DIVs is constructed in various helper routines in order to make this routine easy to read (by humans that is). The various helper routines all are called with a string of space characters; this should improve the the readability of the page that is generated eventually.

The header background changes every N minutes (N=configurable between 0 and 60). Also, the choice of background image is linked to the currently selected item in the main navigation.

Note that the routine `$this->get_div_messages()` does in fact generate its own DIV tags. This is done in order to completely get rid of the message DIV, we do not even want to see an empty DIV if there are no messages.

The same logic applies to the breadcrumb trail.

string function ThemeCornelia::get_menu([\$m = "], [\$menu_id = NULL]) [*line 257*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *int* **\$menu_id** indicates where to start the menu (NULL = first breadcrumb in top level menu)

construct the submenu starting at \$menu_id OR the first breadcrumb in the top level menu

this constructs an 'infinitely' nested set of submenus, starting at \$menu_id or at the first breadcrumb in the top level menu (if any). If there are no suitable nodes, an empty string is returned.

this is largely the same routine as `parent::get_menu()`. Difference is that here we may add a menu title to the menu IF the first item in the breadcrumb trail is a visible section

- Uses [Theme::show_tree_walk\(\)](#)

string function ThemeCornelia::get_quicktop([\$m = "]) [*line 217*]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct a list of quicklinks for top of page (if any) + (maybe) a print-button

- **Uses** [Theme::get_quicklinks\(\)](#)

Package wastheme_frugal Procedural Elements

frugal_install.php

/program/themes/frugal/frugal_install.php -- installer of the frugal theme

This file contains the frugal theme installer. The interface consists of these functions:

```
frugal_install(&$messages,$theme_id)
frugal_upgrade(&$messages,$theme_id)
frugal_uninstall(&$messages,$theme_id)
frugal_demodata(&$messages,$theme_id,$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wastheme_frugal
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: frugal_install.php,v 1.8 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function frugal_demodata(&\$messages, \$theme_id, \$config, \$manifest, \$configuration) [line 292]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

- *array* **\$configuration** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this theme
- **\$config**

add demonstration data to the system

this routine is a no-op because all frugal demodata is already created in the main demodata-routine in /program/install/demodata.php. This routine is retained here as an example and also because a routine by this name should exist (even if it does nothing).

Note If the theme is installed via the Install Wizard, this routine is called. However, if a theme is installed as an additional theme after installation, the `{ $theme }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from admin.php, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct \$config when we're NOT the Instal Wizard.

Fortunately 'frugal' does not need demodata.

bool function frugal_install(&\$messages, \$theme_id) [line 61]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

install the theme

this routine performs the necessary actions to make this theme usable. More specific, this routine adds a handfull of default values into the themes_properties table. Once a theme is actually used in an area, these defaults are copied from the themes_properties table to the themes_areas_properties table for the selected area. The user can subsequently edit these properties in the Area Manager.

bool function frugal_uninstall(&\$messages, \$theme_id) [line 262]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

uninstall the theme

this is a hook for future extensions of Website@School. For now we simply return success. Any real code could look like this:

```
DELETE FROM themes_areas_properties WHERE theme_id = $theme_id; DELETE FROM
themes_properties WHERE theme_id = $theme_id; DELETE FROM themes WHERE
theme_id = $theme_id;
```

or whatever.

bool function frugal_upgrade(&\$messages, \$theme_id) [line 240]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

upgrade the theme

this routine performs an upgrade to the installed theme. Note that this minimalistic 'frugal' theme does not need any upgrade at all because there never was an earlier version.

However, if there was to be a newer version of this theme, this routine is THE place to bring the database up to date compared with the existing version. For example, if an additional property 'foobar' was to be added to the theme configuration, it could be added to the themes_properties table with a suitable (default) value, Any existing areas with this theme could have their configuration updated with this additional foobar property, e.g. INSERT INTO themes_properties: foobar for all areas in themes_areas_properties with theme_id = \$theme_id do INSERT INTO themes_areas_properties: foobar etcetera,

The current version of the theme could be determined by consulting the database (db_select_single_record(themes,'*', 'theme_id = \$theme_id') etcetera.

Note that it is the responsibility of the caller to correctly store the data from the manifest in the themes table. You don't have to do this here, in this routine.

frugal_manifest.php

/program/themes/frugal/frugal_manifest.php - description of the frugal theme

This file defines the frugal theme. This file is used when this theme is installed.

- **Package** wastheme_frugal
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: frugal_manifest.php,v 1.14 2016/06/28 13:47:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/en/frugal.php - translated messages for theme (English)

- **Package** wastheme_frugal
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: frugal.php,v 1.8 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

frugal.php

/program/themes/frugal/languages/nl/frugal.php - translated messages for theme (Nederlands)

- **Package** wastheme_frugal
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: frugal.php,v 1.7 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wastheme_frugal Classes

Class ThemeFrugal

[line 27]

/program/themes/frugal/frugal.class.php - the class that comprises the most minimal theme

- **Package** wastheme_frugal
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: frugal.class.php,v 1.6 2016/03/23 09:28:28 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wastheme_rosalina Procedural Elements

rosalina.php

/program/themes/rosalina/languages/en/rosalina.php - translated messages for theme (English)

- **Package** wastheme_rosalina
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: rosalina.php,v 1.6 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.php

/program/themes/rosalina/languages/nl/rosalina.php - translated messages for theme (Nederlands)

- **Package** wastheme_rosalina
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: rosalina.php,v 1.6 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina.class.php

/program/themes/rosalina/rosalina.class.php - a theme with HV Menu (Javascript-based)

- **Package** wastheme_rosalina
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: rosalina.class.php,v 1.8 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

rosalina_install.php

/program/themes/rosalina/rosalina_install.php -- installer of the rosalina theme

This file contains the rosalina theme installer. The interface consists of these functions:

```
rosalina_install(&$messages,$theme_id)
rosalina_upgrade(&$messages,$theme_id)
rosalina_uninstall(&$messages,$theme_id)
rosalina_demodata(&$messages,$theme_id,$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wastheme_rosalina
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: rosalina_install.php,v 1.6 2016/03/23 09:28:29 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function rosalina_demodata(&\$messages, \$theme_id, \$config, \$manifest) [line 234]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table
- *array* **\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this theme

add demonstration data to the system

this routine adds demonstration data to the freshly installed system

Note If the theme is installed via the Install Wizard, this routine is called. However, if a theme

is installed as an additional theme after installation, the `{ $theme }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

The array `$config` contains the following information.

```
$config['language_key'] => install language code (eg. 'en')
$config['dir']          => path to CMS Root Directory (eg. /home/httpd/htdocs)
$config['www']          => URL of CMS Root Directory (eg. http://exemplum.eu)
$config['progdir']      => path to program directory (eg. /home/httpd/htdocs/program)
$config['progwww']      => URL of program directory (eg. http://exemplum.eu/program)
$config['datadir']      => path to data directory (eg. /home/httpd/wasdata/alb2c3d4e5f6)
$config['user_username'] => userid of webmaster (eg. wblader)
$config['user_full_name'] => full name of webmaster (eg. Wilhelmina Bladergroen)
$config['user_email']   => email of webmaster (eg. w.bladergroen@exemplum.eu)
$config['user_id']      => numerical user_id (usually 1)
$config['demo_salt']    => password salt for all demodata accounts
$config['demo_password'] => password for all demodata accounts
$config['demo_areas']   => array with demo area data
$config['demo_groups']  => array with demo group data
$config['demo_users']   => array with demo user data
$config['demo_nodes']   => array with demo node data
```

With this information, we can add a demonstration configuration for the public area, which shows off the possibilities.

array function `rosalina_get_properties()` [line 388]

construct a list of default properties for this theme

this routine is used when installing rosalina for the first time and also when upgrading the theme to the latest version

Note that HV Menu has a lot of parameters dealing with frames and framesets. Because no frames are used in the Rosalina theme, these parameters are not part of the defaults. However, instead of deleting these settings from this installation script I have commented them out in the array `$properties` below. The necessary values for these 'missing' parameters (which must exist in the JavaScript configuration of HV Menu) are set in the constructor. The effect is that the user never sees these parameters and thus cannot set them to incorrect values for Rosalina.

- Usedby [rosalina_upgrade\(\)](#)
- Usedby [rosalina_install\(\)](#)

bool function `rosalina_install(&$messages, $theme_id)` [line 64]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

install the theme

this routine performs the necessary actions to make this theme usable. More specific, this routine adds a handfull of default values into the themes_properties table. Once a theme is actually used in an area, these defaults are copied from the themes_properties table to the themes_areas_properties table for the selected area. The user can subsequently edit these properties in the Area Manager.

- Uses [rosalina_get_properties\(\)](#)

bool function rosalina_uninstall(&\$messages, \$theme_id) [line 185]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

uninstall the theme

this is a hook for future extensions of Website@School. For now we simply return success. Any real code could look like this:

```
DELETE FROM themes_areas_properties WHERE theme_id = $theme_id; DELETE FROM
themes_properties WHERE theme_id = $theme_id; DELETE FROM themes WHERE
theme_id = $theme_id;
```

or whatever.

bool function rosalina_upgrade(&\$messages, \$theme_id) [line 124]

Function Parameters:

- *array* **&\$messages** collects the (error) messages

- `int $theme_id` the key for this theme in the themes table

upgrade the theme

this routine performs an upgrade to the installed theme.

Note that the initial version of this 'rosalina' theme does not need any upgrade at all because there never was an earlier version (well, duh).

However, if there was to be a newer version of this theme, this routine is THE place to bring the database up to date compared with the existing version. For example, if an additional property 'foobar' was to be added to the theme configuration, it could be added to the themes_properties table with a suitable (default) value, Any existing areas with this theme could have their configuration updated with this additional foobar property, e.g. INSERT INTO themes_properties: foobar for all areas in themes_areas_properties with theme_id = \$theme_id do INSERT INTO themes_areas_properties: foobar etcetera,

The current version of the theme could be determined by consulting the database (db_select_single_record(themes, '*', 'theme_id = \$theme_id') etcetera.

Note that it is the responsibility of the caller to correctly store the data from the manifest in the themes table. You should not do this here, in this routine.

Currently this is a quick and dirty routine to

- update changed sort_order in the existing settings, OR
- add fields that were not available in the current settings

In the future we could make it more sophisticated by updating the themes_areas_properties too. Oh well. KISS

- **TODO** maybe make this a little less quick and dirty?
- **Uses** [rosalina_get_properties\(\)](#)

rosalina_manifest.php

/program/themes/rosalina/rosalina_manifest.php - description of the rosalina theme

This file defines the rosalina theme. This file is used when this theme is installed.

- **Package** wastheme_rosalina
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: rosalina_manifest.php,v 1.10 2016/06/28 13:47:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wastheme_rosalina Classes

Class ThemeRosalina

[line 31]

this class implements the rosalina theme (based on HV Menu)

- **Package** wastheme_rosalina

ThemeRosalina::\$menu_sub

array = [line 37]

- **Var** \$menu_sub holds limits for submenu items (in px): min_width,char_width,max_width,height

ThemeRosalina::\$menu_top

array = [line 34]

- **Var** \$menu_top holds limits for toplevel menu items (in px):
min_width,char_width,max_width,height

Constructor *void* function ThemeRosalina::ThemeRosalina(\$theme_record, \$area_id, \$node_id) *[line 55]*

Function Parameters:

- *array* **\$theme_record** the record straight from the database
- *int* **\$area_id** the area of interest
- *int* **\$node_id** the node that will be displayed

construct a ThemeRosalina object

First we do the regular initialisation, and subsequently we calculate the areas available to this user in `$this->jumps`; Also we set all hvmenu-parameters that are not already set in `$this->config`. This makes it possible to drop certain parameters from the configuration (see [rosalina install\(\)](#)) and still construct a valid hvmenu config.

Finally we pre-calculate the limits `$menu_top` and `$menu_width` for use in the treewalker (see [rosalina show tree walk\(\)](#)).

string function ThemeRosalina::get_html() [*line 137*]

construct an output page in HTML

This constructs a full HTML-page, starting at the DTD and ending with the html closing tag.

string function ThemeRosalina::get_logo([*\$m* = ""]) [*line 259*]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct an image tag with the area logo

This constructs HTML-code that displays the logo, maybe in the form of a clickable map.

This routine honours the `preview_mode` by replacing the URLs with a bare `"#"` in order to prevent the visitor to actually surf away from a preview

- **TODO** should we take `path_info` into account here too???? how about `/area/aaa/node/nnn` instead of `/aaa/nnn`???

string function ThemeRosalina::get_menu_areas([\$m = ""]) [*line 607*]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct a simple UL-based jump menu to select another area (when no Javascript is available)

this constructs a list of clickable links to navigate to other areas. This function is only used when the user has disabled Javascript (it is sandwiched between noscript-tags, see [get_html\(\)](#)). Note that there is no point in having a jump menu when there is not at least another area. If there is only one, an empty string is returned

- **Uses** `$WAS_SCRIPT_NAME`

string function ThemeRosalina::get_quickbottom([\$m = ""]) [*line 635*]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct a list of quicklinks for bottom of page (if any) ending with a separator

This is a slight variation of `parent::get_quickbottom()`: if there is at least one quicklink at the bottom, we append a quickbottom separator to the result in order to visually separate the quicklinks at the left from the appropriate legal notices.

- **Uses** [Theme::get_quicklinks\(\)](#)

string function ThemeRosalina::rosalina_get_page_head([\$m = ""]) [*line 224*]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct the page top

string function ThemeRosalina::rosalina_hotspot_map(\$logged_in, \$mapname, \$hotspots, \$parameters, [\$m = ''])
[line 337]

Function Parameters:

- *bool* **\$logged_in** use the alternative href/title if available
- *string* **\$mapname** name of the HTML-map
- *int* **\$hotspots** the number of hotspots to create
- *array* **\$parameters** an array holding (among other items) the \$hotspots hotspot definition lines
- *string* **\$m** left margin for increased readability

create a hotspot map for the logo

This constructs an HTML map called \$mapname, with \$hotspots shapes linked to the corresponding URLs. Note that this routine honours the the preview_mode by replacing URLs with a bare "#".

The construction of this map is performed by consulting the theme configuration. The shapes are defined via a single string per item, e.g. \$parameters['logo_hotspot_1']. However, note that \$parameters may contain more information than just these shape definitions (as a matter of fact it is simply a copy of the _full_ theme configuration of this area).

A shape definition line should look like this:

```
shape ";" coords ":" href ";" title ";" alt_href  
";" alt_title ";" target
```

where the first three components are mandatory (shape, coords, href) and the rest is optional. As a rule a hotspot is linked to 'href' and 'title'. However, if the user is logged_in, the alternative parameters 'alt_href' and 'alt_title' area used instead. This allows for a single hotspot acting differently based on the logged in status. If no alternative href/title is defined, the standard href/title are used. The last parameter indicates the target of the link. By default it is undefined which implies the same browser window. It could be '_blank' to open in a fresh window. Note that the target parameter only works when NOT in preview mode (otherwise they could still escape from the preview window).

Example: If the definition would be

```
"rectangle;0,0,284,71;/index.php?login=1;Login;/index.php?logout=1;Logout"
```

the hotspot would link to the login-box when the user is NOT logged in, and the user would be logged out when she was already logged in.

string function ThemeRosalina::rosalina_hvmenu([\$m = ""]) [line 472]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct the necessary JavaScript code for definition of HV Menu

this constructs the necessary configuration Array's for HV Menu. The first menu (menu at the top level) can be either Horizontal or Vertical. In the latter case we need to calculate the width of the menu in pixels (#width_px) based on the longest menu item. In the former case every item has its own individual length. This is indicated to the treewalker by setting \$width_px to 0.

- **Uses \$CFG**

string function ThemeRosalina::rosalina_hvmenu_config([\$m = ""]) [line 430]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct the necessary JavaScript that HV Menu needs

the configuration of HV Menu is stored in the theme configuration in \$this->config. Note that the names of the parameters are carefully chosen to match those that HV Menu expects. These parameters can be recognised by the first 7 ASCII-characters in their names, i.e. they all start with 'hvmenu_'.

The PHP variable type determines how we translate the value of these parameters to the correct JavaScript form. The array with pointers to images of arrows Arrws is a special case. The values 'float' parameters get 2 decimals for completeness' sake. Booleans map to integers 1 (True) and 0 (False).

Another exception is the HV Menu-parameter NoOffFirstLineMenus (sic), The number of top level menu items is the number of (visible) items in the tree.

int function ThemeRosalina::rosalina_menucount(\$node_id) [line 585]

Function Parameters:

- *int* **\$node_id** indicates where to start

calculate the number of items in the section (menu) starting at \$node_id

this steps through the linked list (section) starting at \$node_id and simply counts the number of visible items

int function ThemeRosalina::rosalina_menuwidth(\$node_id) [*line 566*]

Function Parameters:

- *int* **\$node_id** indicates where to start

calculate the maximum-width of the items in the section (menu) starting at \$node_id

this steps through the linked list (section) starting at \$node_id and determines the maximum width of the link text of visible items, expressed in characters (not bytes)

string function ThemeRosalina::rosalina_navigation_menu([\$m = ""]) [*line 374*]

Function Parameters:

- *string* **\$m** left margin for increased readability

construct the navigation menu

This implements the HV Menu by Ger Versluis.

- **Uses \$CFG**

string function ThemeRosalina::rosalina_show_tree_walk([\$m = "], \$subtree_id, \$menu_name, \$width_px) [*line 509*]

Function Parameters:

- *string* **\$m** left margin for increased readability

- *int* **\$subtree_id** starting node of this menu
- *string* **\$menu_name** base name of the corresponding JavaScript menu
- *int* **\$width_px** the precalculated width of this menu OR 0 if individual widths wanted (only at top level)

this treewalker shows the current menu and descends recursively

this routine creates a menu and descends into any submenus (sections)

Note that we expect the caller to pre-calculate the width of the menu items (in \$width_px). However, if \$width_px is 0, we calculate individual widths per item, using the limits from \$this->menu_top, because only [rosalina_hvmenu\(\)](#) can call us with \$width_px = 0 and thus we are creating the top level menu. Any submenus are constructed by recursion and this routine never calls itself with \$width_px = 0.

Package wastheme_ruta Procedural Elements

ruta.php

/program/themes/ruta/languages/en/ruta.php - translated messages for theme (English)

- **Package** wastheme_ruta
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: ruta.php,v 1.2 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

ruta.php

/program/themes/ruta/languages/nl/ruta.php - translated messages for theme (Nederlands)

- **Package** wastheme_ruta
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: ruta.php,v 1.2 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

ruta_install.php

/program/themes/ruta/ruta_install.php -- installer for the ruta theme

This file contains the ruta theme installer. The interface consists of these functions:

```
ruta_install(&$messages,$theme_id)
ruta_upgrade(&$messages,$theme_id)
ruta_uninstall(&$messages,$theme_id)
ruta_demodata(&$messages,$theme_id,$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wastheme_ruta
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: ruta_install.php,v 1.1 2016/03/08 19:29:07 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function ruta_demodata(&\$messages, \$theme_id, \$config, \$manifest) [*line 423*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table
- *array* **\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this theme

add demonstration data to the system

this routine adds demonstration data to the freshly installed system

Note If the theme is installed via the Install Wizard, this routine is called. However, if a theme

is installed as an additional theme after installation, the `{ $theme }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

The array `$config` contains the following information.

```
$config['language_key'] => install language code (eg. 'en')
$config['dir']          => path to CMS Root Directory (eg. /home/httpd/htdocs)
$config['www']          => URL of CMS Root Directory (eg. http://exemplum.eu)
$config['progdir']      => path to program directory (eg. /home/httpd/htdocs/program)
$config['progwww']      => URL of program directory (eg. http://exemplum.eu/program)
$config['datadir']      => path to data directory (eg. /home/httpd/wasdata/alb2c3d4e5f6)
$config['title']        => the name of the site (as entered by Wilhelmina Bladergroen)
$config['user_username'] => userid of webmaster (eg. wblader)
$config['user_full_name'] => full name of webmaster (eg. Wilhelmina Bladergroen)
$config['user_email']   => email of webmaster (eg. w.bladergroen@exemplum.eu)
$config['user_id']      => numerical user_id (usually 1)
$config['demo_salt']    => password salt for all demodata accounts
$config['demo_password'] => password for all demodata accounts
$config['demo_areas']   => array with demo area data
$config['demo_groups']  => array with demo group data
$config['demo_users']   => array with demo user data
$config['demo_nodes']   => array with demo node data
$config['demo_string']  => array with demo strings from /program/install/languages/LL/demodata.php
$config['demo_replace'] => array with search/replace pairs to 'jazz up' the demo strings
```

With this information, we can add a demonstration configuration for the public area, which shows off the possibilities.

bool function ruta_install(&\$messages, \$theme_id) [line 61]

Function Parameters:

- *array &\$messages* collects the (error) messages
- *int \$theme_id* the key for this theme in the themes table

install the theme

perform the necessary actions to make this theme usable; add a handful of default values into the `themes_properties` table. Once a theme is actually used in an area, these defaults are copied from the `themes_properties` table to the `themes_areas_properties` table for the selected area. The user can subsequently edit these properties in the Area Manager.

bool function ruta_uninstall(&\$messages, \$theme_id) [line 371]

Function Parameters:

- *array &\$messages* collects the (error) messages

- *int* **\$theme_id** the key for this theme in the themes table

uninstall the theme

this is a hook for future extensions of Website@School. For now we simply return success. Any real code could look like this:

```
DELETE FROM themes_areas_properties WHERE theme_id = $theme_id; DELETE FROM
themes_properties WHERE theme_id = $theme_id; DELETE FROM themes WHERE
theme_id = $theme_id;
```

or whatever.

bool function ruta_upgrade(&\$messages, \$theme_id) [*line 350*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

upgrade the theme

this routine performs an upgrade to the installed theme. It does nothing (for now), but it must exist.

ruta_manifest.php

/program/themes/ruta/ruta_manifest.php - description of the ruta theme

This file defines the ruta theme. It is used when this theme is installed.

- **Package** wastheme_ruta
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: ruta_manifest.php,v 1.2 2016/06/28 13:47:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wastheme_ruta Classes

Class ThemeRuta

[line 27]

/program/themes/ruta/ruta.class.php - the class that implements the theme

- **Package** wastheme_ruta
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: ruta.class.php,v 1.1 2016/03/08 19:29:07 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

ThemeRuta::\$ruta_header_background

string = [line 30]

- **Var** the additional style that implements the rotating banner background

Constructor *void* function ThemeRuta::ThemeRuta(\$theme_record, \$area_id, \$node_id) [line 40]

Function Parameters:

- *array* **\$theme_record** the record straight from the database
- *int* **\$area_id** the area of interest

- *int* **\$node_id** the node that will be displayed

construct a ThemeRuta object

- See [Theme::Theme\(\)](#)

string function ThemeRuta::alt_show_tree_walk([\$m = "], \$subtree_id, \$max_level) [*line 756*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *int* **\$subtree_id** indicates where to start this (sub)menu
- *int* **\$max_level** maximum nesting displayed if not breadcrumbs

workhorse for constructing alternative recursive menu (walk the tree)

construct nested (sub)menus for a depth of at most \$max_level, unless we hit the breadcrumb trail in which case we carry on nesting. The (sub)menus are constructed in the form of nested UL's with LI's.

The level of recursion of the list items (LI) is indicated via class='levelNNN'. The type of item is indicated via class='page' or class='section'. Also, the item has an additional class='current' when it is part of the breadcrumb trail. Finally a current item also has the additional class 'activepage' or 'activesection' which makes the CSS easier.

The actual A-tag of the link only indicates being part of the breadcrumb trail via class='current'.

It is up to the style sheet to visualise these items taking all variants into account. Note that we only process visible pages and sections.

Note: this routine looks very much like {@see show_tree_walk()}. The difference is the limit \$max_level.

string function ThemeRuta::get_altnavigation([\$m = "], [\$textonly = FALSE]) [*line 709*]

Function Parameters:

- *string* **\$m** left margin for increased readability

- **bool \$textonly** forces a text-type link even when a navigation image is stipulated

create alternative navigation in case Javascript is not available

This creates separated DIVs, one for each item in the main navigation bar, each containing the corresponding submenu. This whole alternative navigation DIV is set to display: none using Javascript. If there is no Javascript, it remains visible. The submenus are displayed for up to 2 levels deep, unless the current breadcrumb trail leads deeper into the tree. This way most of the important nodes are reachable.

string function ThemeRuta::get_bazaar_style_style([\$m = ""]) [*line 616*]

Function Parameters:

- *string* **\$m** margin for increased readability

insert special rotating background banner in bazaar style style in header

the background style is precomputed once and stored in \$this->ruta_header_background. This extends the original routine in such a way that the calculated background is inserted *_before_* the used-defined BSSS. This means that the user can overrule the background image information if she wants to.

string function ThemeRuta::get_bottomline([\$m = ""]) [*line 269*]

Function Parameters:

- *string* **\$m** left margin for increased readability

show footer text, maybe some quicklinks and 'powered by'

void function ThemeRuta::get_html() [*line 115*]

string function ThemeRuta::get_menu([\$m = "], [\$menu_id = NULL]) [*line 237*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *int* **\$menu_id** indicates where to start the menu (NULL = first breadcrumb in top level menu)

construct the submenu starting at \$menu_id OR the first breadcrumb in the top level menu

this constructs an 'infinitely' nested set of submenus, starting at \$menu_id or at the first breadcrumb in the top level menu (if any). If there are no suitable nodes, an empty string is returned.

this is largely the same routine as parent::get_menu(). Difference is that here we may add a menu title to the menu IF the first item in the breadcrumb trail is a visible section

- **Uses** [Theme::show_tree_walk\(\)](#)

string function ThemeRuta::get_menunavigation([\$m = "], [\$textonly = FALSE]) [*line 675*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *bool* **\$textonly** forces a text-type link even when a navigation image is stipulated

construct a top level menu (navigation bar)

walk through the top level of the menu tree and create a link for each node. This variant of the top level navigation is to be displayed under the submenu in the left hand margin in case the screen is small (<650). In that case the main navigation bar is limited to 'MENU' and 'HOME' and the rest of the main navigation has to be available elsewhere. Different entries here are all members of the class 'menunavsub'.

string function ThemeRuta::get_navigation([\$m = "], [\$textonly = FALSE]) [*line 636*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *bool* **\$textonly** forces a text-type link even when a navigation image is stipulated

construct a top level menu (navigation bar) with an additional menu button

start with a menu-button (that can be made invisible via CSS) and subsequently walk through the top level of the menu tree and create a link for each node. the first real link gets a different class so we can keep the first real link visible while hiding the others via CSS. This is

used to create a navbar with two links 'MENU' and 'HOME' on the small (<650px) screen.

Note that this MENU-button uses a tiny snippet of Javascript to toggle the menu.

string function ThemeRuta::ruta_get_background(\$index, [\$fully_qualified = FALSE], [\$m = ""]) [*line 502*]

Function Parameters:

- *int* **\$index** indicates current main navigation item or 0 for none
- *bool* **\$fully_qualified** TRUE forces URLs with scheme, authority etc.
- *string* **\$m** margin added for readability

compute background image(s) based on current time and index

this routine returns a snippet of ready-to-use code to set a background image for #header. The images are different depending on the time of day and the \$index parameter. The latter is thought to be the position of the current item in the main navigation (starting at 1) or 0 in case no item in the main navigation is current.

Configuration is done via two parameters: the paths in \$this->config['header_banners_directories'] (one per line) and the interval in \$this->config['header_banners_interval'].

The interval is expressed in minutes. It means that a particular page has the same background image for the duration of the interval and another one in the next interval. The banners directories are supposed to contain banner files of the correct dimensions, e.g. 650x71 (small), 980x170 (medium) or 1440x240 (large).

If the specified directory starts with a '/' it is assumed to start in the data root, e.g. something like /areas/exemplum/banners. This will eventually lead to a URL of the form /file.php/areas/exemplum/banners/mentha-banner.jpg using was_file_url()

If the specified directory does NOT start with a '/' it is assumed to be a directory relative to the directory where index.php and friends live, unless it starts with 'program/' in which case it is a static dir somewhere in the program hierarchy. This is done via was_url().

Note: if files are found in a directory, the names are sort()ed so there is a chance that the three files are the same ones independent of the physical order in the file system directory.

- Uses [was_url\(\)](#)

- Uses [was_file_url\(\)](#)
- Uses [utf8_validate\(\)](#)

string function ThemeRuta::ruta_get_sidebar(\$sidebar_nodes, [\$m = ""]) [*line 411*]

Function Parameters:

- *array* **\$sidebar_nodes** contains node-module pairs to show
- *string* **\$m** add to human readability

retrieve data for sidebar

this retrieves the content of the various sidebar blocks (if any) currently only the htmlpage module is supported.

string function ThemeRuta::ruta_get_sidebar_htmlpage(\$index, \$node_id, [\$m = ""]) [*line 444*]

Function Parameters:

- *in* **\$index** block number (1-based)
- *int* **\$node_id** identifies the pagedata to retrieve
- *string* **\$m** increased readability

retrieve page data for htmlpage module on a node

this retrieves data from the page \$node_id (requires knowledge of internals of htmlpage module/table) in order to show that in a box in the 3rd column. The content is wrapped in a div which has a unique ID of the form sidebar_blockN where N counts from 1 upward and a common class sidebar_htmlpage. This allows for connecting style information to a particular sidebar block in a flexible way.

int function ThemeRuta::ruta_navigation_index() [*line 290*]

calculate the index of the current main navigation item

this routine determines which main menu item is the current one, starting at 1. if no menu item is current, the function returns 0.

This index is used for two purposes:

- calculating an ever changing background image for the header, and
- determining which item of the comma-delimited pagenumbers to use for the sidebar.

bool/array function ThemeRuta::ruta_sidebar_nodes_modules(\$index) [*line 343*]

Function Parameters:

- *int* **\$index** is the current main menu item or 0 for none

compute the list of sidebar blocks for a main menu item

this constructs an ordered array of node_id's corresponding to the current main menu item. If the current main menu_item indicates no sidebar then FALSE is returned.

The basis for the node_ids to return is the contents of the parameter 'sidebar_nodelist'. This is a comma-delimited string containing node_ids, zeros or dashes.

Example: suppose sidebar_nodelist contains "5,0,0,0,-". If the first item of the main menu is current (ie. is part of the breadcrumb trail), the node with node_id = 5 is examined. If it is a page and the module is supported, this page's node_id is returned. If, however, this is a section, all supported pages within that section are returned. For main menu items 2, 3 and 4 a '0' is specified. This means that no pages will be displayed (an empty array is returned). For main menu item 5 a dash '-' is specified. This yields a return value of FALSE which must be interpreted as 'suppress the complete 3rd column of the page, including rightmargin_top and rightmargin_bottom. This effectively reduces the 3-column layout to a 2-column layout.

Note that the node_ids in the sidebar_nodelist themselves must not be under embargo but a hidden section (with visible pages) is OK.

Note that when there are more main menu items than entries in the sidebar_nodelist, the value '0' is assumed.

Finally, if no main menu item is current (eg. the visitor arrived on a page via quicktop entries) then the value of '0' is assumed, too.

string function ThemeRuta::show_menu_js([\$m = ""]) [*line 799*]

Function Parameters:

- **\$m**

insert code for toggling the menu/free HTML in the left hand margin

Package wastheme_schoolyard Procedural Elements

schoolyard.php

/program/themes/schoolyard/languages/en/schoolyard.php - translated messages for theme (English)

- **Package** wastheme_schoolyard
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: schoolyard.php,v 1.7 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.php

/program/themes/schoolyard/languages/nl/schoolyard.php - translated messages for theme (Nederlands)

- **Package** wastheme_schoolyard
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: schoolyard.php,v 1.6 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard.class.php

/program/themes/schoolyard/schoolyard.class.php - implements the Schoolyard Theme by David Prousch

- **Package** wastheme_schoolyard
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: schoolyard.class.php,v 1.8 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

schoolyard_install.php

/program/themes/schoolyard/schoolyard_install.php -- installer of the schoolyard theme

This file contains the schoolyard theme installer. The interface consists of these functions:

```
schoolyard_install(&$messages,$theme_id)
schoolyard_upgrade(&$messages,$theme_id)
schoolyard_uninstall(&$messages,$theme_id)
schoolyard_demodata(&$messages,$theme_id,$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wastheme_schoolyard
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: schoolyard_install.php,v 1.5 2016/03/23 09:28:30 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function schoolyard_demodata(&\$messages, \$theme_id, \$config, \$manifest) [*line 234*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table
- *array* **\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this theme

add demonstration data to the system

this routine adds demonstration data to the freshly installed system

Note If the theme is installed via the Install Wizard, this routine is called. However, if a theme

is installed as an additional theme after installation, the `{ $theme }_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

The array `$config` contains the following information.

```
$config['language_key'] => install language code (eg. 'en')
$config['dir']          => path to CMS Root Directory (eg. /home/httpd/htdocs)
$config['www']          => URL of CMS Root Directory (eg. http://exemplum.eu)
$config['progdir']      => path to program directory (eg. /home/httpd/htdocs/program)
$config['progwww']      => URL of program directory (eg. http://exemplum.eu/program)
$config['datadir']      => path to data directory (eg. /home/httpd/wasdata/alb2c3d4e5f6)
$config['user_username'] => userid of webmaster (eg. wblader)
$config['user_full_name'] => full name of webmaster (eg. Wilhelmina Bladergroen)
$config['user_email']   => email of webmaster (eg. w.bladergroen@exemplum.eu)
$config['user_id']      => numerical user_id (usually 1)
$config['demo_salt']    => password salt for all demodata accounts
$config['demo_password'] => password for all demodata accounts
$config['demo_areas']   => array with demo area data
$config['demo_groups']  => array with demo group data
$config['demo_users']   => array with demo user data
$config['demo_nodes']   => array with demo node data
```

With this information, we can add a demonstration configuration for the public area, which shows off the possibilities.

array function `schoolyard_get_properties()` [line 305]

construct a list of default properties for this theme

this routine is used when installing schoolyard for the first time and also when upgrading the theme to the latest version

- Usedby [schoolyard_upgrade\(\)](#)
- Usedby [schoolyard_install\(\)](#)

bool function `schoolyard_install(&$messages, $theme_id)` [line 64]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

install the theme

this routine performs the necessary actions to make this theme usable. More specific, this routine adds a handfull of default values into the themes_properties table. Once a theme is actually used in an area, these defaults are copied from the themes_properties table to the themes_areas_properties table for the selected area. The user can subsequently edit these properties in the Area Manager.

- Uses [schoolyard_get_properties\(\)](#)

bool function schoolyard_uninstall(&\$messages, \$theme_id) [*line 185*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

uninstall the theme

this is a hook for future extensions of Website@School. For now we simply return success. Any real code could look like this:

```
DELETE FROM themes_areas_properties WHERE theme_id = $theme_id; DELETE FROM  
themes_properties WHERE theme_id = $theme_id; DELETE FROM themes WHERE  
theme_id = $theme_id;
```

or whatever.

bool function schoolyard_upgrade(&\$messages, \$theme_id) [*line 124*]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

upgrade the theme

this routine performs an upgrade to the installed theme.

Note that the initial version of this 'schoolyard' theme does not need any upgrade at all because there never was an earlier version (well, duh).

However, if there was to be a newer version of this theme, this routine is THE place to bring the database up to date compared with the existing version. For example, if an additional property 'foobar' was to be added to the theme configuration, it could be added to the themes_properties table with a suitable (default) value. Any existing areas with this theme could have their configuration updated with this additional foobar property, e.g. INSERT INTO themes_properties: foobar for all areas in themes_areas_properties with theme_id = \$theme_id do INSERT INTO themes_areas_properties: foobar etcetera,

The current version of the theme could be determined by consulting the database (db_select_single_record(themes, '*', 'theme_id = \$theme_id') etcetera.

Note that it is the responsibility of the caller to correctly store the data from the manifest in the themes table. You should not do this here, in this routine.

Currently this is a quick and dirty routine to

- update changed sort_order in the existing settings, OR
- add fields that were not available in the current settings

In the future we could make it more sophisticated by updating the themes_areas_properties too. Oh well. KISS

- **TODO** maybe make this a little less quick and dirty?
- **Uses** [schoolyard_get_properties\(\)](#)

schoolyard_manifest.php

/program/themes/schoolyard/schoolyard_manifest.php - description of the schoolyard theme

This file defines the schoolyard theme. This file is used when this theme is installed or upgraded.

- **Package** wastheme_schoolyard
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: schoolyard_manifest.php,v 1.10 2016/06/28 13:47:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wastheme_schoolyard Classes

Class ThemeSchoolyard

[line 30]

this class implements the schoolyard theme (based on a design by David Prousch)

- **Package** wastheme_schoolyard

Constructor *void* function ThemeSchoolyard::ThemeSchoolyard(\$theme_record, \$area_id, \$node_id) [line 42]

Function Parameters:

- *array* **\$theme_record** the record straight from the database
- *int* **\$area_id** the area of interest
- *int* **\$node_id** the node that will be displayed

construct a ThemeSchoolyard object

First we do the regular initialisation, and subsequently we set a few parameters (very un-exciting).

string function ThemeSchoolyard::get_html() [line 71]

construct an output page in HTML

This constructs a full HTML-page, starting at the DTD and ending with the html closing tag.

This routine returns a full HTML-page, usually including logo, (area) title, main navigation,

breadcrumbs trail (optional), menu, jumpmenu and a footer with links to printer friendly version + logout. If the page is called with `print=1` as one of the get-parameters, all those extra stuff is suppressed by including the additional `print.css` stylesheet (configurable). This allows for making a clean print of only the content. This additional stylesheet is added only once, even if this routine is called more than once (shouldn't happen). This stylesheet is configurable just like the regular stylesheet.

Note that there might be a jumpmenu (to go to other areas). This is only displayed if there is another area to go to. If the current area is the only one available, we don't bother the user with an extra navigation widget.

string function ThemeSchoolyard::schoolyard_get_div_quicktop([\$m = ""]) [line 230]

Function Parameters:

- *string* **\$m** margin for readability

construct an optional div for quicklinks at the top if any

string function ThemeSchoolyard::schoolyard_logout([\$separator = "], [\$m = ""]) [line 209]

Function Parameters:

- *string* **\$separator** a visual separator that is prepended
- *string* **\$m** margin for increased readability

conditionally construct a logout link

string function ThemeSchoolyard::schoolyard_printpage([\$separator = "], [\$m = ""]) [line 185]

Function Parameters:

- *string* **\$separator** a visual separator that is prepended
- *string* **\$m** margin for increased readability

construct a 'print this page' link

Package wastheme_sophia Procedural Elements

sophia.php

/program/themes/sophia/languages/en/sophia.php - translated messages for theme (English)

- **Package** wastheme_sophia
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: sophia.php,v 1.4 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

sophia.php

/program/themes/sophia/languages/nl/sophia.php - translated messages for theme (Nederlands)

- **Package** wastheme_sophia
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: sophia.php,v 1.4 2016/03/23 09:28:32 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

sophia_install.php

/program/themes/sophia/sophia_install.php -- installer for the sophia theme

This file contains the sophia theme installer. The interface consists of these functions:

```
sophia_install(&$messages,$theme_id)
sophia_upgrade(&$messages,$theme_id)
sophia_uninstall(&$messages,$theme_id)
sophia_demodata(&$messages,$theme_id,$config,$manifest)
```

These functions can be called from the main installer and/or admin.php.

Note You cannot be sure about the environment from which these routines are called. If the caller is the Install Wizard, you do not have all subroutines available. However, it IS possible to manipulate the database via the `db_*`() routines and/or the global `$DB` object. Therefore you have to keep the install routine extremely simple. You also have no option to interact with the user; the install has to be a silent install; you can only indicate success (TRUE) or failure (FALSE) and maybe an error message in `$messages[]` but that's it. Good luck.

- **Package** wastheme_sophia
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: sophia_install.php,v 1.5 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

bool function sophia_demodata(&\$messages, \$theme_id, \$config, \$manifest) [line 359]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table
- *array* **\$config** pertinent data for the new website + demodata foundation
- *array* **\$manifest** a copy from the manifest for this theme

add demonstration data to the system

this routine adds demonstration data to the freshly installed system

Note If the theme is installed via the Install Wizard, this routine is called. However, if a theme

is installed as an additional theme after installation, the `{$_theme}_demodata()` routine is never called. This is because the only time you know that demodata is installed is when the Install Wizard runs. If we're called from `admin.php`, the webmaster may have already deleted existing (core) demodata so you never can be sure what to expect. To put it another way: it is hard to re-construct `$config` when we're NOT the Instal Wizard.

The array `$config` contains the following information.

```
$config['language_key'] => install language code (eg. 'en')
$config['dir']           => path to CMS Root Directory (eg. /home/httpd/htdocs)
$config['www']           => URL of CMS Root Directory (eg. http://exemplum.eu)
$config['progdir']       => path to program directory (eg. /home/httpd/htdocs/program)
$config['progwww']       => URL of program directory (eg. http://exemplum.eu/program)
$config['datadir']       => path to data directory (eg. /home/httpd/wasdata/alb2c3d4e5f6)
$config['title']         => the name of the site (as entered by Wilhelmina Bladergroen)
$config['user_username'] => userid of webmaster (eg. wblader)
$config['user_full_name'] => full name of webmaster (eg. Wilhelmina Bladergroen)
$config['user_email']    => email of webmaster (eg. w.bladergroen@exemplum.eu)
$config['user_id']       => numerical user_id (usually 1)
$config['demo_salt']     => password salt for all demodata accounts
$config['demo_password'] => password for all demodata accounts
$config['demo_areas']    => array with demo area data
$config['demo_groups']   => array with demo group data
$config['demo_users']    => array with demo user data
$config['demo_nodes']    => array with demo node data
$config['demo_string']   => array with demo strings from /program/install/languages/LL/demodata.php
$config['demo_replace']  => array with search/replace pairs to 'jazz up' the demo strings
```

With this information, we can add a demonstration configuration for the public area, which shows off the possibilities.

bool function sophia_install(&\$messages, \$theme_id) [line 61]

Function Parameters:

- *array &\$messages* collects the (error) messages
- *int \$theme_id* the key for this theme in the themes table

install the theme

this routine performs the necessary actions to make this theme usable. More specific, this routine adds a handful of default values into the `themes_properties` table. Once a theme is actually used in an area, these defaults are copied from the `themes_properties` table to the `themes_areas_properties` table for the selected area. The user can subsequently edit these properties in the Area Manager.

bool function sophia_uninstall(&\$messages, \$theme_id) [line 307]

Function Parameters:

- *array &\$messages* collects the (error) messages

- *int* **\$theme_id** the key for this theme in the themes table

uninstall the theme

this is a hook for future extensions of Website@School. For now we simply return success. Any real code could look like this:

```
DELETE FROM themes_areas_properties WHERE theme_id = $theme_id; DELETE FROM
themes_properties WHERE theme_id = $theme_id; DELETE FROM themes WHERE
theme_id = $theme_id;
```

or whatever.

bool function sophia_upgrade(&\$messages, \$theme_id) [line 285]

Function Parameters:

- *array* **&\$messages** collects the (error) messages
- *int* **\$theme_id** the key for this theme in the themes table

upgrade the theme

this routine performs an upgrade to the installed theme. Note that this minimalistic 'sophia' theme does not need any upgrade at all because there never was an earlier version.

However, if there was to be a newer version of this theme, this routine is THE place to bring the database up to date compared with the existing version. For example, if an additional property 'foobar' was to be added to the theme configuration, it could be added to the themes_properties table with a suitable (default) value, Any existing areas with this theme could have their configuration updated with this additional foobar property, e.g. INSERT INTO themes_properties: foobar for all areas in themes_areas_properties with theme_id = \$theme_id do INSERT INTO themes_areas_properties: foobar etcetera,

The current version of the theme could be determined by consulting the database (db_select_single_record(themes, '*', 'theme_id = \$theme_id') etcetera.

Note that it is the responsibility of the caller to correctly store the data from the manifest in the themes table. You don't have to do this here, in this routine.

sophia_manifest.php

/program/themes/sophia/sophia_manifest.php - description of the sophia theme

This file defines the sophia theme. It is used when this theme is installed.

- **Package** wastheme_sophia
- **Author** Peter Fokker < peter@berestijn.nl>
- **Version** \$Id: sophia_manifest.php,v 1.8 2016/06/28 13:47:26 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Package wastheme_sophia Classes

Class ThemeSophia

[line 27]

/program/themes/sophia/sophia.class.php - the class that implements the theme

- **Package** wastheme_sophia
- **Author** Peter Fokker < peter@berestijn.nl >
- **Version** \$Id: sophia.class.php,v 1.4 2016/03/23 09:28:31 peter Exp \$
- **Copyright** Copyright (C) 2008-2016 Ingenieursbureau PSD/Peter Fokker
- **License** [GNU AGPLv3+Additional Terms](#)

Constructor *void* function ThemeSophia::ThemeSophia(\$theme_record, \$area_id, \$node_id) [line 29]

Function Parameters:

- **\$theme_record**
- **\$area_id**
- **\$node_id**

string function ThemeSophia::get_bottomline([\$m = ""]) [line 208]

Function Parameters:

- *string* **\$m** left margin for increased readability

show footer text, maybe some quicklinks and 'powered by'

string function ThemeSophia::get_html() [*line 63*]

construct an output page in HTML

This constructs a full HTML-page, starting at the DTD and ending with the html closing tag.

The page is constructed using nested DIVs, the layout is taken care of in a separate style sheet. All knowledge about the structure of the page is contained in this routine.

The performance of the script (# of queries, execution time) is calculated as late as possible, to catch as much as we can. Therefore the construction is done in two parts and performance is calculated last.

The contents of the various DIVs is constructed in various helper routines in order to make this routine easy to read (by humans that is). The various helper routines all are called with a string of space characters; this should improve the the readability of the page that is generated eventually.

Note that the routine `$this->get_div_messages()` does in fact generate its own DIV tags. This is done in order to completely get rid of the message DIV, we do not even want to see an empty DIV if there are no messages.

The same logic applies to the breadcrumb trail.

string function ThemeSophia::get_menu([\$m = "], [\$menu_id = NULL]) [*line 176*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *int* **\$menu_id** indicates where to start the menu (NULL = first breadcrumb in top level menu)

construct the submenu starting at \$menu_id OR the first breadcrumb in the top level menu

this constructs an 'infinitely' nested set of submenus, starting at \$menu_id or at the first breadcrumb in the top level menu (if any). If there are no suitable nodes, an empty string is returned.

this is largely the same routine as `parent::get_menu()`. Difference is that here we may add a menu title to the menu IF the first item in the breadcrumb trail is a visible section

- Uses [Theme::show_tree_walk\(\)](#)

string function ThemeSophia::get_navigation([\$m = "], [\$textonly = FALSE]) [*line 141*]

Function Parameters:

- *string* **\$m** left margin for increased readability
- *bool* **\$textonly** forces a text-type link even when a navigation image is stipulated

construct a top level menu (navigation bar) as an unnumbered list (UL) of list items (LI)

this walks through the top level of the menu tree and creates a link for each node with a distinct class for every LI This allows for buttons in different shapes and colours.

By default the theme comes with three different shapes (see also style.css):

- an ellipse (file: stencil0.png, 110x70)
- a hexagon (file: stencil6.png, 110x70)
- a drum (file: stencil8.png, 110x70)

Different colours are assigned (in style.css) by overruling the default colour using the (unique) class of each button. Look for classes "navigation_buttonN", with N=1,2,...

Appendices

Appendix A - Class Trees

Package wascore

AclManager

- [AclManager](#)

AdminOutput

- [AdminOutput](#)

AdminSkin

- [AdminSkin](#)

AlertManager

- [AlertManager](#)

AreaManager

- [AreaManager](#)

ConfigAssistant

- [ConfigAssistant](#)

DatabaseMysql

- [DatabaseMysql](#)

DatabaseMysqli

- [DatabaseMysqli](#)

DatabaseMysqliResult

- [DatabaseMysqliResult](#)

DatabaseMysqliResult

- [DatabaseMysqliResult](#)

Email

- [Email](#)

FileManager

- [FileManager](#)

GroupManager

- [GroupManager](#)

Language

- [Language](#)

PageManager

- [PageManager](#)

Theme

- [Theme](#)
 - [ThemeAxis](#)
 - [ThemeCornelia](#)
 - [ThemeFrugal](#)
 - [ThemeRosalina](#)
 - [ThemeRuta](#)
 - [ThemeSchoolyard](#)
 - [ThemeSophia](#)

TranslateTool

- [TranslateTool](#)

Useraccount

- [Useraccount](#)

UserManager

- [UserManager](#)

Zip

- [Zip](#)

Zip

- [Zip](#)

Package default

CKEditor

- [CKEditor](#)

CKEditor

- [CKEditor](#)

CrewClient

- [CrewClient](#)

CrewServer

- [CrewServer](#)

CrewWorkshop

- [CrewWorkshop](#)

FCKeditor

- [FCKeditor](#)

CKEditor

- [CKEditor](#)

Package waslang_ar

Package waslang_bg

Package waslang_da

Package waslang_de

Package waslang_el

Package waslang_fa

Package waslang_en

Package waslang_es

Package waslang_fi

Package waslang_fr

Package waslang_hi

Package waslang_hu

Package waslang_it

Package waslang_nl

Package waslang_pl

Package waslang_pt

Package waslang_ro

Package waslang_sh

Package waslang_ru

Package waslang_sk

Package waslang_sv

Package waslang_tr

Package waslang_ur

Package waslang_vi

Package waslang_zh

Package waslang_fil

Package waslang_fry

Package waslang_pap

Package wasinstall

InstallWizard

- [InstallWizard](#)

Package wasmod_crew

Package wasmod_guestbook

Package wasmod_snapshots

SnapshotViewer

- [SnapshotViewer](#)
 - [SnapshotViewerInline](#)

Package wasmod_htmlpage

Package wasmod_sitemap

Package wasmod_mailpage

Package wasmod_confab

Package wasmod_mypage

Package wasmod_search

Package wasmod_althing

Package wasmod_newsletter

Package wasmod_aggregator

Package wasmod_redirect

Package wastheme_axis

ThemeAxis

- [Theme](#) (Different package)
 - [ThemeAxis](#)

Package wastheme_ruta

ThemeRuta

- [Theme](#) (Different package)
 - [ThemeRuta](#)

Package wastheme_schoolyard

ThemeSchoolyard

- [Theme](#) (Different package)
 - [ThemeSchoolyard](#)

Package wastheme_frugal

ThemeFrugal

- [Theme](#) (Different package)
 - [ThemeFrugal](#)

Package wastheme_cornelia

ThemeCornelia

- [Theme](#) (Different package)
 - [ThemeCornelia](#)

Package wastheme_rosalina

ThemeRosalina

- [Theme](#) (Different package)
 - [ThemeRosalina](#)

Package wastheme_sophia

ThemeSophia

- [Theme](#) (Different package)
 - [ThemeSophia](#)

Appendix B - README/CHANGELOG/INSTALL

FAQ

/program/FAQ.txt

\$Id: FAQ.txt,v 1.1.1.1 2011/02/01 13:00:05 pfokker Exp \$

This file will contain answers to frequently asked questions.

Q1: Can I temporarily close my website?

A1: Yes. If a file called 'maintenance.html' exists in the top level directory (i.e. the directory holding the main entry points and also the file 'config.php'), a website visitor will be redirected to that file 'maintenance.html' instead of the real site.

Q2: What is this message "condition code 010"?

A2: It means that the site is not yet configured. Point your browser to /program/install.php to start the installation wizard and follow the instructions.

Q3: What is "condition code 050"?

A3: This indicates that the version of the program (the .PHP-files) is not the same as the structure of the database. Usually this means that an update needs to be done in the database, changing the database structure.

TODO

/program/TODO.txt

\$Id: TODO.txt,v 1.4 2014/11/07 13:22:19 peter Exp \$

2008-01-29

- convert /program/*.txt to DOS-text
- decide on exact license conditions

2008-02-01

- add an explanation to the manual about \$CFG->datadir within the document root and a 'difficult' name that cannot be guessed.

2008-02-07

OK should we document \$CFG->debug? this is a feature that is handy during development, but it should be 'off' on a production server
Answer: this is now documented.

- how can we discriminate between a call from the command line and a call via a web interface? Is it a mere check of \$_SERVER['REMOTE_ADDR']? (You _never_ have a remote address when working from the command line, do you?). Would be handy in case of cron.php.

2008-02-14

-- make a separate subtree for testing purposes. I don't think that that test environment should be published for end users (but it should be for developers). Maybe a subdirectory /devel/test/ and a corresponding /devel/test/test.php?

2008-02-19

-- can it be possible to have a page visible in navigation and not available to serve? I'm inclined to say yes: if a node only has a href, no actual data is available in the database. However, should we treat this as a call nevertheless and count the page view just before we redirect via header()? Mmm....

2008-03-22

OK Protect password fields with AUTOCOMPLETE="off" (see http://www.owasp.org/index.php/Guide_to_Authentication)

OK Count the number of failed login-attempts within a certain period of time

OK On-time password reset should time out in a short time, e.g. 15 minutes (ok, make it 30 minutes)

OK Should there be password requirements (min 1 digit, min 1 capital, minimum length?): YES one of each.

OK Add delay in user validation (3 seconds per attempt)

OK Should login-forms have short-lived unique identifiers? (prevents brute force attacks?) Methinks that our blacklist facility (8 minutes after more than 10 failed attempts) is enough.

2008-04-25

-- Add to cronjobs: cleanup of expired sessions + logging of the fact

OK Should we add the number of calls + length of session in seconds to the logout message? What then if the session expires without a proper logoff? A: added this for regular logouts

-- Add to cron: removal of obsolete login_failures? It should be auto-cleaning but if a user fails once and never returns, the failure will be there forever.

OK Get rid of the 'trigger_error()' calls now that we have logger(), maybe except errors in the database routines..

-- Logrotate for log_messages table? How? Send a summary to the webmaster before deletion?

2008-04-27

-- we should be able to optimise with relative links (sometimes) if the hostname part of \$CFG->www and \$CFG->progwww are the same. Example: when pointing to the source of an image the src would be {\$CFG->progwww}/graphics/foo.png, i.e. including the 'http://www.exemplum.websiteatschool.org' part. If \$CFG->www also starts with that string, we can leave it out in the src property: src="/program/graphics/foo.png".

Here is a generic example from RFC3986:

```
foo://example.com:8042/over/there?name=ferret#nose
  \_/  \_____/  ^  \_____/  \_____/  \_____/
  |         |         |         |         |
scheme authority path query fragment
```

If \$CFG->www and \$CFG->progwww have the same scheme and authority, links could be made relative, e.g.
 and

 instead of prefixing with an explicit "foo://example.com:8042/". What if the scheme and authority are NOT the same?

2008-05-01

-- About the installer: if the installer is not able to write 'config.php' to the target directory, it should be possible to 'download' a .ZIP-file with this config.php, and perhaps also with index.php, cron.php, admin.php? Unzip that in the appropriate place and you are in business.

2008-08-01

-- How about storing ETags and be friendly to caches/proxies?
-- How about using index.php/aaa/nnn/friendly-title-of-page-for-bookmark instead of index.php?area=aaa&node=nnn
That too is cache/proxy-friendly...

2008-09-29

OK How about adding an extra parameter to manual.php in order to 'deep link' to the manual? Example: if the user has selected the pagemanager in admin.php, should the help button be calling 'manual.php?language=en&job=pagemanager' or something? Done: additional parameter 'topic' is added, at least from page manager. Could be employed from start centre too if necessary.

2008-10-01

-- Should there be an attribute 'is_root' or 'is_guru' in the users table? E.g. if a node is owned by some user, only that user is allowed to remove the read-only bit. Shouldn't there be a way to override that other than by using phpMyAdmin to manipulate the database directly?

2008-10-10

-- Edit permissions not only depend on the user's permission bits but also on the read-only attribute of a node.
-- We should isolate the check-permission routines because they are used when creating icons and also when executing the corresponding function.

2008-10-20

-- get some decent graphics for button_save and button_cancel and other buttons...

2008-10-28

OK How about changing all where clauses with "(value = 'NULL') into "(value IS NULL)" to be more SQL-standard compliant? Done, in databaselib.php and function db_where_clause().
-- So how about recovering from a crashed browser with an inaccessible session? It is very annoying to be locked out in the cold if the browser crashed and I immediately login again using the same credentials and still not be able to carry on...
Should we add the time of the lock, too? That would give the other user an educated guess about the odds that the other session is still active. Mmmmm...

2008-11-12

-- the output object AdminOutput should have something like a 'funnel mode'. This mode should disable all navigation links and other

links that could distract and/or seduce the user to leave the page. Main use: prevent locked records hanging around without the user actively editing that record.

2008-11-20

- the Theme should have a property 'date/time last modified' so a module can update it via `module_view()`.
- how to create an alert based on edit content (or not)? How can we tell that the content actually changed? Is there a 'dirty' bit somewhere? Should we return a status instead of simply true/false?

2008-12-10

- How about the visibility of nodes when in preview mode????

2009-02-25

- do we actually `_USE_` the field `muser_id` in the page manager? we do in the area manager.

2009-02-26

- How about adding transactions (rollback, commit)
See function that deletes areas.
Q: How well is MySQL suited for transactions?
A: Mmmmm.... Which version? Which storage engine?

2009-05-31

- Why not use the 'name' field in the `dialogdef[]` arrays?
`$dialogdef = array('fullname' => array('name' => 'fullname', 'type' => F_STRING, ...));` Much more convenient when saving data.
- Doublecheck for `htmlspecialchars()` when displaying information from the database in a dialog: a username like say '`<u>ser</u>name`' should be converted to '`<u>ser</u>name`' before being sent to the browser.

2009-06-04

- we should re-analyse the repercussions of the option to delete user accounts. If only the `user_id` is logged, it is difficult to figure out whodunnit if the account has already disappeared, not to mention breaking the constraint of the foreign key. This is also a problem for tables with `cuser_id` and `muser_id`...

2009-09-22

- We should have a graphics designer restyle all icons etc. I'm no good at that.

2009-12-04

- Should we make the permissions 0700 configurable, eg in `areamanager`?

2010-09-27

- There is a potential security issue with relative paths: the check on `'../'` is inconclusive if the `$path` is encoded in UTF-8: the overlong sequence `2F C0 AE 2E 2F` eventually yields `2F 2E 2E 2F` or `'../'`. Reference: RFC3629 section 10. Applies to `main_file.php` and also `filemanager.class.php`.

2010-12-08

-- Should we add a configurable favicon to themes as a config option (defaulting to /program/graphics/favicon.ico)?

2011-05-02

-- Can we think of a smart way to assign hotkeys to HTML-labels where the hotkey is a multi-byte UTF-8 sequence? And more important: does it make sense to do so or is a hotkey like [Alt-R] always connected to the keyboard-key 'R' in combination with 'Alt'? Does it even work to have something like "~\xC2\xAEegister" where \xC2\xAE is the (R) symbol and the tilde would indicate hotkey [Alt-(R)] or what? And, how does one generate such a character (R)? If you generate it with say [AltGr] + [R] then the hotkey-combination becomes [Alt-AltGr-R]. Mmmmm, sounds counter-intuitive to me.

-- Is it better to assign quasi-random ASCII-characters to translated labels, something like "~<A>\xC2\xAEegister" which might yield hotkey [Alt-A] for the label that would display as "\xC2\xAEegister" ie without the '~' '<' 'A' and '>'. Mmmm.... confusing...

2011-09-09

-- How about normalising all calls to _SERVER['REMOTE_ADDR'] making sure we can compare 'canonical' IP-addresses (see RFC 5952).

LICENSE

/program/LICENSE.txt

\$Id: LICENSE.txt,v 1.2 2011/02/03 14:04:01 pfokker Exp \$

The 'LICENSE AGREEMENT for Website@School' can be found in the file 'license.html' in the program directory or on-line at <<http://websiteatschool.eu/license.html>>.

The necessary 'Appropriate Legal Notices' can be found in the file 'about.html' in the program directory.

[eof]

README

/program/README.txt

\$Id: README.txt,v 1.3 2016/06/28 13:21:16 peter Exp \$

This is the README for the Website@School Content Management System.

See INSTALL.txt for installation instructions.
See LICENSE.txt for license information.
See HISTORY.txt for the project history.
See CREDITS.txt for information about contributors to Website@School.
See FAQ.txt for answers to frequently asked questions.
See CHANGES.txt for significant changes between revisions.
See TODO.txt for a list of things that still need to be done.

The homepage of the Website@School Project is at <<http://websiteatschool.eu>>.

HISTORY

/program/HISTORY.txt
\$Id: HISTORY.txt,v 1.1.1.1 2011/02/01 13:00:01 pfokker Exp \$

History of Website@School

=====

Website@School started in Amsterdam in 2002 and as they say: the rest is history.

INSTALL

/program/INSTALL.txt
\$Id: INSTALL.txt,v 1.5 2016/06/28 13:21:16 peter Exp \$

Quick Install of Website@School

=====

1. Download the latest distribution version of Website@School (either as a .ZIP-file or a tar.gz-file.), e.g. from <<http://download.websiteatschool.eu>>.
2. Unpack the distribution file somewhere on your webserver, e.g. in /home/httpd/htdocs. This will yield a number of files (index.php, admin.php, etc.) in the CURRENT directory. Also, a subdirectory 'program' with all the other program files will be created.
3. (optional but recommended) Install the Website@School Manual:
 - 3A. Download the latest version of the Website@School Manual (either as a .ZIP-file or a tar.gz-file.), e.g. from <<http://download.websiteatschool.eu>>.
 - 3B. Unpack the manual file in the same place as you did unpack the distribution file in step 2 above.

4. Start the installation process by pointing your browser to 'program/install.php' and follow the directions.
5. Make sure that cron.php is called regularly (e.g. via cronjob).
6. Your new website is ready.

Quick Upgrade of Website@School

=====

1. Download the latest distribution version of Website@School (either as a .ZIP-file or a tar.gz-file.), e.g. from <<http://download.websiteatschool.eu>>.
2. Unpack the distribution file in the same place on your webserver as you did before, e.g. in /home/httpd/htdocs. This will yield a number of files (index.php, admin.php, etc.) in the CURRENT directory. Also, a subdirectory 'program' with all the other program files will be created. Note: the new version is supposed to overwrite the existing version.
3. (optional but recommended) Install or update the Website@School Manual, by downloading the latest version and unpacking the file in the same directory you used before (see step 2).
4. Surf to the admin.php file and login. If you are not redirected to the Update Manager please do so manually, using the Tools menu.
- 5A. A clickable link in the Status column indicates that the core system needs to be updated. Please follow that link. After this update the status should change to 'OK'.
- 5B. Please follow the other links (if any) in the Status column until every item has the status 'OK'.
6. Your website is now up to date.

More information

=====

For more information and detailed installation instructions please consult the Website@School Manual. The latest version of the manual can be found on-line at <<http://manual.websiteatschool.eu>>.

[eof]

CHANGES

/program/CHANGES.txt

\$Id: CHANGES.txt,v 1.324 2016/06/28 14:38:47 peter Exp \$

2016-06-29

Finally: Website@School 1.0.0 is released!

This release is dedicated to Dirk Schouten, the Website@School Project Leader, who passed away on June 29, 2015, after a short illness.

For many years Dirk has devoted virtually all of his time and energy to the project: dreaming up ideas for new modules, finding donors for funding, coordinating translator activities, and writing the comprehensive Website@School manual. His efforts were recognised when he was appointed Member in the Order of Orange-Nassau by her Majesty the Queen of The Netherlands in 2012. Dirk was and will remain an inspiration to us all. He will be greatly missed.

2016-06-28

- import/update of all translations
- new translations (work in progress): Hindi, Slovak.
- preparations for the upcoming new release

2016-06-16

- changes in demodata: replaced the placeholder for 'news' with an instance of the aggregator; replaced placeholders for newsletter and newsletter archive with redirects to newsletter in showcase
- modernised HTML-headers in install.php

2016-06-15

- new feature: statistics: overview of pageviews, per area or for all areas

2016-06-14

- added to the CREDITS.txt and donors.php: two donors

2016-06-02

- imported the original mime.types that was the basis of the mapping between file extensions and mime types in filelib.php. This file was retrieved from <https://svn.apache.org/viewvc/httpd/httpd/trunk/docs/conf/mime.types?revision=104298&view=co> Original date: 2004-07-15.
- imported up-to-date mime.types file from <https://svn.apache.org/viewvc/httpd/httpd/trunk/docs/conf/mime.types?revision=1742054&view=co> Original date: 2016-05-03. According to the header this file has been placed in the public domain for unlimited distribution. [devel/import/mime.types].
- quick and dirty tool mimemerge.php for merging mime.types into filelib.php
- re-arranged the logic that checks filename extensions and -mimetypes in filelib.php, filemanager.class.php main_file.php (and also in newsletter_admin.php)

2016-05-26

- moved queue_area_node_alert() from PageManager to waslib.php for general use
- new feature: we now send alerts when a new post to an althing is submitted
- new feature: we now send alerts when a new newsletter article is submitted
- bugfix: althing moderation never yielded an alert

2016-05-18

- new feature: Session Tool allows for killing existing sessions, to be used when a user's browser has crashed or accidentally closed and a page/section was still locked.
- new permission in Admin for Session Tool (value 2048)

2016-05-17

- remove FK constraint on locked pages/sections from table 'nodes';
- bump internal version from 2015041100 to 2016051700

2016-04-26

- new feature: alert manager: add/edit/delete alerts

2016-04-19

- added context-sensitive help link for alerts
- added context-sensitive help link for theme ruta

2016-04-18

- bump internal version from 2015033100 to 2016041100 after adding favicon

2016-04-13

- added download option for log message table [toolslib.php]
- added prune function to log message viewer [toolslib.php]

2016-04-11

- added support for favicon.ico in main site configuration

2016-04-10

- Upgraded CKEditor from 4.4.5 to 4.5.8 (latest and greatest), including both the full source file and the already smaller, stripped sources in was/devel/imports

2016-04-08

- addition to snapshots module: the slideshow is now interactive; navigation is now possible with cursor keys and spacebar

2016-04-06

- final version of search module

2016-04-05

- all 12 modules now have a working interface for search:
 - 2 x NOP: mypage, redirect
 - 8 x 1-on-1: aggregator, confab, crew (workshop), htmlpage, mailpage, search, sitemap, snapshots
 - 2 x 1-on-N: althing, newsletter

2016-04-04

- added two search pages to global demodata
- added search support to newsletter module
- bugfix: redacted posts could be seen in full by specifying ?post=n [althing]

2016-04-01

- new module: search
- lots of changes in all other modules to accomodate search
- minor tweaks

2016-03-23

- both profile and password dialog require additional authorisation to prevent passers by changing a user's profile [mypage]

2016-03-22

- new module: mypage
- added new module mypage to demodata too
- small changes in hotkeys in admin.php because ~D is now [~Done]

- converted login dialogs to responsive html 5 (mobile first)

2016-03-21

- moved `get_skin_names()` and `get_editor_names()` from `usermanager.class.php` to `waslib.php` in order to reuse that code in the MyPage module
- added redirect to global `$USER` for easier editing of profile in MyPage

2016-03-08

- updated `license.html` and `about.html` to be more responsive too
- removed old-fashioned HTML-comment trick for `STYLE` tags [`theme.class.php`]
- new theme Ruta

2016-02-29

- theme Frugal is completely responsivified both in changed code and style information [`base.css`, `theme.class.php`]

2016-02-23

- removed meta `http-equiv Content-Style-Type` from `theme.class.php` because `text/css` is the default anyway
- moved `Content-Script-Type` from meta `http-equiv` to `http` headers although it may be a little superfluous nowadays
- keep `http-equiv Content-Type` so `charset UTF-8` is obvious when document is at rest
- removed `http-equiv MSSmartTagsPreventParsing` altogether
- removed the old-fashioned comment/script transition
- "description" and "keywords" are now user-editable instead of hard-coded in `theme.class.php`
- cosmetics
- added `CFG->language_key` to `html` root tag

2016-02-22

- added `newsletter.html` to context-sensitive help [`manual.php`]

2015-06-16

- bugfix: there was an error in the access control: some permissions did not cascade properly [`pagemanager.class.php`]

2015-05-29

- new module: newsletter

2015-04-23

- bugfix: `dbsession_remove_obsolete_sessions()` violated a FK constraint; we now first release the lock and subsequently drop the session record.

2015-04-03

- removed calls to `cron_send_queued_alerts()` from `main_admin.php` and `pagemanager.class.php` because we now have a 'real' cron function. This needs to be setup though: `/cron.php` should be called periodically for this to work smoothly (say: at least once per day), preferably every hour.

2015-04-02

- re-arranged code in `main_file.php` to easily check user file availability

2015-03-31

- added support for cron jobs: two new options in site configuration
- `cron.php` now actually works and calls `$module_cron` for appropriate modules

2015-03-30

- funnel-mode is causing headaches combined with pagination; we now no longer enforce funnel mode on the pagination feature.

2015-03-08

- added functionality to Email class:
 - . we can now handle multiple alternative message bodies automatically
 - . it is now possible to add bot plain and related attachments
 - . there is now a function to prepare a full message body without actually sending it

2015-02-06

- add support for mysqli interface:
 - . new classes DatabaseMysqli and DatabaseMysqliResult [mysqli.class.php]
 - . mysql/mysqli-specific code in Install Wizard

2015-02-03

- bugfixes:
 - . typo in wasentry_script_name() [init.php, install.php]
 - . replaced deprecated eregi() and ereg_replace() with preg_match and preg_replace() [translatetool.class.php, email.class.php, [toolslib.php]
 - . renamed password_hash() to was_password_hash() to prevent nameclashes [loginlib.php, usermanager.class.php]
 - . re-arranged function call to stop warning about pass by reference [theme.class.php, filemanager.class.php]

Release 0.90.6a - 2014-11-17 / 2014111700

2014-11-14

- showstopper in upgrade routine: wrong database engine for additional tables. Fixed in mysql.class.php.
- Release 0.90.6 is NOT published.

Release 0.90.6 - 2014-11-11 / 2014111100

2014-11-11

- minor tweaks in various translations
- added to the CREDITS.txt: translators, donors and others
- final preparations for new release

2014-11-10

- Added 9 new languages: fil, fry, it, pap, ro, sh, vi, bg and ur.
- Updated available translations for the other languages

2014-11-09

- Demodata for Aggregator is now complete (i.e. with demo snapshots too)
- Minor tweaks

2014-11-08

- Added logo + link for four new donors:
 - . Stichting Blindenhulp
 - . Landelijke Stichting voor Blinden en Slechtienden
 - . Rotterdamse Stichting Blindenbelangen
 - . Stichting tot Verbetering van het Lot der Blinden
- Added demodata for Workshop (CREW) as much as possible.
- Bugfix: no longer a hard-coded clear:both in aggregator pages
- Partial demodata for Aggregator added.

2014-11-07

- Added CKEditor 4.4.5 too (latest and greatest), including both the full source file and the already smaller, stripped sources in was/devel/imports
- Added option 'ckeditor3' to list of available editors in user manager and site configuration, including 12 translations.
- Core version bumped due to change in config; forces update to new version.
- Added refresh button to confab moderation NoMessage dialog
- Added a contact form to the showcase in the demodata

2014-11-06

- Updated CKEditor 3.6.2 -> 3.6.6.2.
- Removed the old source file from devel, added new one instead.

2014-11-03

- re-arranged moderation dialog; split into two parts [confab]
- some code rearranged [confab]
- up to date translations in confab
- added special wide class to admin_base.css for use in confab moderation
- all accesskeys are now in upper case [dialoglib.php]
- added download button to confab report
- added full sitemap to showcase
- added redirect to website@school to showcase

2014-10-30

- added moderation option to confab
- added style support for zebra-type listings to admin_base and admin_lowvision
- added Dutch translation of Confab
- minor tweaks and typos in other translations

2014-10-29

- added report selections to report a la Althing [confab_admin.php]

2014-10-28

- added hotkeys to general login dialogs [loginlib.php]
- added hotkeys to the Confab join dialog too [confab_view.php]

2014-10-26

- added dynamic resizing to Visual interface to make the best use of available screen real estate
- Visual interface is now complete (in 4 different CSS-flavours)
- added support for minimised version visual.min.js in makedist.sh

2014-10-24

- more progress with Visual interface: automatic updates now work with XMLHttpRequest.

2014-10-22

- some progress with the Visual interface but still unfinished

2014-10-21

- added new variable to class Theme: \$silent_mode: allows modules to suppress the regular \$theme output if set to TRUE [theme.class.php]
- added access keys to the input fields in Talk-screen [confab_view.php]
- done with the Braille interface

2014-10-19

- bugfix; we now force our own session_expiry timeout [dbsession.php]
- yet another confab-field added to database: message_from_id can keep track of participant colour display (attributes) in Visual interface

2014-10-16

- more or less done with the Braille interface
- code cleanup [confab_view.php]
- started with Visual interface

2014-10-14

- basic functionality of Braille interface now works

2014-10-09

- we now also accept 'friendly parameters' in login [main_index.php]
- it is now possible to gain access to the Confab via the passcode
- Confab login works

2014-10-08

- we no longer show the uneditable conversation_id in config dialog [config_admin.php]
- added fields to keep track of userlist updates in Confab
- minor cosmetics

2014-10-06

- added get_cookie_string() to validate/retrieve cookie in waslib.php

2014-10-02

- reporting function in confab added
- minor cosmetics and typos

2014-10-01

- added confab-specific demodata, currently visible via 'report' (work in progress...)

2014-09-29

- start of confab module

2014-09-16

- a small refinement in the error message display in loginlib dialogs: we now show the word 'Message:' and the message in bold for users who have style switched off (graceful degradation).
- we now attempt to logout the user in case of an error_exit() [init.php]
- we now switch to a new area whenever one is specified in Page Manager. Furthermore, as a service, we open up the path to the specified node (if any) setting the tree view to custom. In combination with the login-feature that propagates the parameters, we can now send a 'deep link' by mail and let a user login and navigate to the correct place within the tree of an area. Handy for tending to moderations in the Althing module.

2014-09-12

- many, many changes in loginlib.php. Highlights:
 - . we now are able to propagate path_info GET[] parameters during login;
 - . the dialog boxes have all been styled via program/styles/login.css;
 - . we no longer use HTML tables for layout (makes for a much nicer view with styling switched off, also more friendly for the visual impaired);
 - . we now recognise the correct path while (re)setting session cookies, allowing multiple instances of was running simultaneous in sibling directories on the same domain;
 - . processing of POSTed credentials now also checks for utf8 validity

- (mostly by employing `validate_dialogdef()`);
- . garbage collection is also performed after succesfull change of password;
- . we now set the focus to input fields near the end of the body rather than via `onload()` in the body tag;
- . added a few relevant keywords for this application via meta-tags;
- . many cosmetic changes and code cleanup.
- correct cookie path via `session_set_cookie_params()` in `main_index.php`, `main_admin.php` and `main_file.php`
- new feature in `index.php`: we can now login on-the-fly on our way to an intranet which requires valid username/password. If the area is still not available (non-existig or no permissions), the infamous error 080 is returned after all.

2014-09-09

- new parameter for text input fields: `autocomplete="off"` [`dialoglib.php`]

2014-09-05

- Bugfix: the `Theme::get_address()` routine did not properly handle magic quotes in `$_GET[]` and `$_SERVER[]`; fixed.
- Bugfix: the `AdminOutput::get_address()` had the same issue: also fixed.
- Two `get_address()` routines replaced with a single function `get_page_address_url()` in `waslib.php`, keeping wrappers for compatibility.
- Removed translation for 'URL' because `get_page_address_url()` no longer prefixes the address with that translation.
- Bugfix: a stupid typo in `html_a()` in `htmllib.php`.

2014-07-25

- added another parameter to the view-routine of the Althing-module: we can now show an individual post from the Althing via parameter 'post'. This is used in the alert e-mail messages.

2014-07-21

- Another change in the User Interface: we now show the module nickname next to the `page_id`, but only if user has edit permissions for that page. There is room for improvement here.
- Added nicknames for all existing modules in English, Dutch.

2014-7-18

- Change in User Interface of Page Manager: now the title of a section opens/closes the section rather than opening the basic properties dialogue. Considerans: once a section exists it is far more likely that the section will be opened/closed rather than edited. One can always use the pencil icon to go to the edit dialogue anyway.
- Another change in the User Interface of Page Manager: in the treeview all pages (not sections) now have a (nick)name for the module associated with that page. This nickname is translatable and it could be empty. Also, in the title (mouseover) the full name of the module is now visible.

2014-07-17

- workaround for an issue in `utf8validate()`: PHP crashes if the string to test (via a Regular Expression) is too long; implemented an alternative algorithm for longer strings [`utf8lib.php`]

2014-07-14

- workaround for known issue in `file(1)` with .css-files [`filelib.php`]
- added .css as an allowable extension to installer [`tabledata.php`]

and also to the updater [updatelib.php]. We can now upload .CSS-files to the datadirectory.

2014-07-11

- added multiple file upload adding the "multiple" attribute in the File Manager; it is now possible to select multiple files in a single file input widget
- added automatic resizing of uploaded images to more convenient via a configurable dimension.
- added new site configuration option: `resize_dimension` (default: 800px). Includes update of existing installations (core version bumped)
- various changes and additions in translation texts, notably in File Manager (add a file) and Site Configuration.

2014-07-02

- bugfix in redirect-routine in `main_admin.php`

2014-06-13

- framing error: changed 'blog' to 'weblog' (blog is weblog without the "we")
- added debugging info in generated BSSS per area/section/page

2014-06-08

- minor cosmetics in Althing moderation: it is no longer an error to have an empty remark if nothing (marbles, visibility) changed anyway
- also added a feedback message in the case that nothing was changed
- minor cosmetics in Mailpage: we now have the same regime as Althing for Cancel buttons, etc.

2014-06-06

- added Dutch translation of Althing (a lot of work)
- returned main menu in demo data to Search (get rid of 'Find')
- changed the MyPage-page into a section as a modules showcase with MyPage now as the first page in that section
- bugfix in updatelib: the options of the althing were not set in an upgrade/install of the module.

2014-06-05

- added real pictures to Althing demo
- minor tweaks in Althing demo data & code
- re-arranged modules showcase: now under Find (previously Search) | Modules

2014-06-04

- added context-sensitive help for modules, themes [`manual.php`]
- changed dimensions of the help pop-up from 640x480 to 760x570 and also added scrollbars and resize option [`main_admin.php`]

2014-06-01

- Althing reporting tool done. The Althing module is now feature complete.

2014-05-31

- started with report function: first draft is working more or less

2014-05-30

- still more tweaks in Althing: the post header now is completely configurable
- re-arranged the default post header layout
- added graphics for navigation up/down in Althing overview
- the link after submitting a post now actually works (if the post is published)

- shortened some translations

2014-05-29

- minor tweaks in Althing
- added debug-parameter to InstallWizard via `install.php?debug=1`
- minor changes in althing demodata, tweaks in althing language
- added sort and show/hide option in althing overview

2014-05-28

- done with althing alerts: we now send alerts too when a post is published by a moderator.
- minor tweaks & cosmetics

2014-05-27

- minor cosmetics in `althing_admin.php`: cleanup email lists
- save new post in `althing_view.php`
- added new file with common constants & routines: `althing_common.php`
- cosmetics in add a post dialog: toggle to show/suppress BBCode help
- added sending of alerts for 6 of the 8 cases: this really works
- bbcode help can now be made visible/hidden by clicking the title
- still work in progress, but almost there (thankyou-screen)

2014-05-26

- althing: work in progress:
 - . add a post dialog works (including link to File Manager for registered users)
 - . detailed explanation of available BBCode
 - . validation of new posts (including BBCode syntax check)

2014-05-20

- various tweaks to the demodata, showing off BBCode-conversion to HTML
- althing overview is done, including styling via a separate CSS-file
- preparations for creating a form to add a post are underway, but still: work in progress

2014-05-19

- some code added for BBCode-handling: rendering & validating
- better display of Althing messages in `althing_view()` (still work in progress)
- small enhancement: user language now also displays language key in the edit user dialog [`usermanager.class.php`]
- `althing_view_show_post()` is still work in progress

2014-05-16

- new feature: if a page/section is locked by another session the user can forcefully unlock (and relock) the page/section if the `user_id`'s of both sessions are the same. This solves the problem of crashing browsers that leave nodes locked until the crashed session times out (which might take too long to be user friendly). [`pagemanager.class.php`, `waslib.php`]
- althing: removed loooong explanations from submenu items; we do have a manual for that
- moderation details screen: a few changes to win some screen real estate:
 - . moderations: xx is and marbles are now on a single line
 - . all radio buttons are now on a single line
 - . the remark-field is limited to a single line
- any empty post header lines (in the translation) are not displayed. the layout of the post details is now completely configurable via the Translate Tool.
- added a preliminary 4th submenu option 'report' to Althing
- removed even more cruft from the various submenu dialogs

- Cancel in all submenu dialogs now return to the Althing Content dialog

2014-05-13

- added \$fragment-parameter to html_a() in htmllib.php
- basically done with the admin-interface of the Althing, only a few stubs are left (and also the Dutch translation). Next step: althing_view().

2014-05-12

- Althing: change in the tabledefinitions: we now recognise three different values for initial visibility: 0=hidden, 1=registered users visible, anonymous cowards hidden, 2=visible. Althing configuration dialog is adjusted
- added the post moderation dialog, including a complex interaction between two sets of two radiobuttons that together identify (and allow the moderator to change) the publication status and the visibility. Short explanation: you cannot UNpublish a post that has been visible, ever, even if it was only for a very short while.
- added more sort options to the moderation list dialog: we can now also sort on the three visibility status-values: visible/hidden/unpublished or vice versa.
- we also show the mtime in a (sortable) column: this shows the date of last modification/moderation.
- added yet another demo-posting from an angry anonymous coward in reaction to the moderation action that Maria did earlier that night. This example/demo is beginning to get realistic.
- removed the clickable date because it cluttered the screen; you can now zoom in on a particular post by clicking the ID in the 1st column
- the sort order is retained while in the moderation detail screen, which makes that the links 'Next' and 'Previous' follow the chosen sort order.

2014-05-11

- added some more demodata and also created special permissions for members of the Seniors group so they can manage the demo-althing.

2014-05-10

- minor tweaks in althing_admin.php
- added a preliminary set of demodata to Althing module including translation
- added another field to the althings table to calculate display_id only once

2014-05-09

- minor cosmetics in Page Manager
- new mailpage is now complete, including the Dutch translation
- removed the submenu (mis)feature from Page Manager; we now have the new module interface with submenu options and all.

2014-05-08

- many changes in mailpage addresses add/edit/delete but still work in progress

2014-05-07

- we now always show the edit menu when editing content via the interface <module>_show_edit() and <module>_save(), even though it may distract the user (just a little).

2014-05-06

- obsoleted file htmlpage.class.php (long overdue)
- some more code cleanup in htmlpage-module
- added CSRF-prevention and [Done] button module Sitemap

- added CSRF-prevention and [Done] button module Redirect
- added CSRF-prevention and [Done] button module Snapshots
- added CSRF-prevention and [Done] button module Aggregator
- added CSRF-prevention and [Done] button module CREW
- added CSRF-prevention and [Done] button module Mailpage (work in progress)
- cosmetics: a little more space in tabular dialogs in the ACL-Mmanager.

2014-05-05

- undone two changes in `show_menu_groupcapacity()` and `show_menu_group()` in `groupmanager.class.php` because it added 'clutter'
- added CSRF-prevention to 7 dialogs in PageManager
- change in datadefinition: added field options to modules table
- changed module interface: added extra options field in `<module>_show_edit()` and `<module>_save()`.
- updated `install.php` to handle new modules options while reading the manifest
- bumped internal version to 2014050500; this forces an upgrade which alters the database adding the options field to the module table [`version.php`]
- modified the html-page module to work with the new module interface
- added CSRF-protection and [Done] button to htmlpage-module
- added two options to mailpage manifest, including translations in modules' language file (all very much work in progress)
- added preliminary update function for existing mailpages (adding two options to module table; see `mailpage_upgrade()` in `mailpage_install.php`).
- a little code cleanup in the htmlpage module

2014-05-01

- added CSRF-prevention to all 4 dialogs in File Manager
- added CSRF-prevention to all 3 dialogs in Translate Tool
- added CSRF-prevention to all 3 dialogs in ACL Manager
- added [Done] button to all 3 dialogs in ACL Manager
- added CSRF-prevention to all dialogs in UserManager
- added [Done] button to all dialogs in UserManager
- added CSRF-prevention to all dialogs in GroupManager
- added [Done] button to all dialogs in GroupManager
- added a link to the group's 'basic properties' in `show_menu_groupcapacity()` in order to make navigation easier between group/capacities and basic group properties.
- added a link to the 'Groups' overview in `show_menu_group()` in order to make navigation easier between basic group properties and the groups overview.

2014-04-30

- added utility routines to prevent CSRF [`dialoglib.php`, `waslib.php`]
- added CSRF-prevention to:
 - . Area Manager
 - . Theme Configuration in AreaManager (via ConfigAssistant)
 - . Site configuration (via ConfigAssistant)
 - . Module manager (via ConfigAssistant)
- added thousands separators in upload dialog [`filemanager.class.php`].
- added [Done] button to upload dialog too
- added [Done] button to create directory [`filemanager.class.php`]

2014-04-29

- added a new standard button 'done' in preparation of a massive change of the way most dialogs in W@S will behave: the current [Save] button will leave the user in the (saved and refreshed)

dialog whereas the new [Done] button will save the data and subsequently leave the dialog. This is the answer to a feature request for 'save + edit again' notably in modules and also in the Theme configuration. This addition brings a lot of small changes in many source files and the addition of a button_done.gif icon file.

- various dialogs in PageManager now have three buttons: [Save] [Done] [Cancel]: add page/section, edit page/section basic, edit page/section advanced. This includes 'following' the user when a page, section or small subtree is moved to another area and the user presses [Save]. (If she moves a node to another area and presses [Done] she stays in the 'old' area.
- Still to come: add [Done] buttons to page Content dialogs (via module interface).
- Added the [Done] button to various other dialogs throughout the system:
 - . AreaManager
 - . Theme Configuration in AreaManager (via ConfigAssistant)
 - . Site configuration (via ConfigAssistant)
 - . Module manager (via ConfigAssistant)

2014-04-25

- new feature in AdminOutput: we now also have a submenu to add to (\$output->add_submenu()). The submenu is simply appended to the menu (at the end) allowing a module to add menu items below the main menu with 'Basic'. 'Advanced' and 'Content'. [main_admin.php]

2014-04-17

- bugfix: we can now create tables without any keys definition in the tabledef [mysql.class.php:480]

2014-01-28

- bugfix: wrong language domains caused untranslated menu-items in new languages (without \$progdire/install/languages/\$LL dir) under install

Release 0.90.5 - 2013-07-11 / 2013071100

2013-07-11

- final tweaks before release (updatelib.php:update_core_20130711())

2013-07-10

- minor tweaks in various stylesheets (input styling)
- preparations for more tweaks in styling added [dialoglib.php]

2013-07-09

- all buttons now also have a class 'button' [dialoglib.php]
- re-arranged styling for skins to use 'button' class
- modified all default styles for existing themes to add styling for input elements
- minor changes in existing skins too, also with button style

2013-07-03

- re-arranged location of contact page in demodata
- added a demo of the redirect module (combined with contact page)
- extra parameter 'friendly_url' now also available in demodata()
- better failsafe loading of translations in install (always load 'en' too)
- added Dutch translations for demo of mailpage in demodata()
- minor cosmetics

2013-07-02

- almost done with the mailpage module including Dutch translation.
- added demodata for the mailpage module in quicklinks (top)

2013-07-01

- additional validation in mailpage email address
- we now suppress the destination listbox if there's only a single option

2013-06-29

- functional version of mailpage_view()

2013-06-28

- small improvements in buttons [dialoglib.php]
- interim-version of mailpage_view()

2013-06-27

- finished with the 'tokens' for visitor forms; added tokenlib.php
- temporarily increased version number to force database update for tokens

2013-06-26

- added new table 'tokens' to keep track of instances of visitor's forms
- prepared for core update routine update_core_2013mddxx() [updatelib.php]

2013-06-20

- added mailpage module
- done with the admin-part of the mailpage module already

2013-06-14

- added new language Greek (el)
- added new language Finnish (fi)
- added new language Russian (ru)
- added new language Swedish (sv)
- updated all existing translations (based on distribution version 0.90.4)

2013-06-13

- minor tweaks in crew readme.txt

2013-06-12

- tweaks in the CREW module
- added a tool to generate minimised javascript code [devel/tools/minjs.php]
- added code to create crew.min.js in crew module [devel/tools/makedist.sh]
- crewserver.zip unpacks in a subdir automatically [devel/tools/makedist.sh]
- the 'running code download' also unpacks in subdir [crewserver.php]

2013-06-11

- bumped year in copyright messages where necessary
- added special code to make a downloadable crewserver.zip
in /program/modules/crew/crewserver.zip [devel/tools/makedist.sh]

2013-06-10

- fixed a few typos in the CREW module
- corrected the message color in Theme Frugal [base.css]

2013-06-06

- added support for maximum # of shops and # of workers/shop via error
code 1008: this disables all buttons etc. except [Cancel] [crew.js]

2013-06-04

- added an (arbitrary) limit on document size: 65536 characters [crew_view.php]
- done with a preliminary version of CREW module (but no websocket server yet)

2013-06-03

- added missing `html_tag_close()` [htmllib.php]
- added new standard button Edit [dialoglib.php and style/translation files]
- new utility function `hmac()` according to RFC2104 [waslib.php]
- added Dutch translation for CREW-module

2013-05-31

- typos in crew module
- simplified the intro text for crew in module manager
- added a few `has_module_*`() permission functions [useraccount.class.php]
- rearranged internal structure of cached ACLs in Useraccount

2013-05-30

- added preliminary version of crew module (Collaborative Remote Educational Workshop)

2013-05-29

- added helptopic 'modulemanager' [manual.php]
- implemented module manager [modulemanagerlib.php]

2012-11-01

- new theme: Cornelia

2012-07-03

- noscript-addition to SnapshotViewerInline (graceful degradation)
- new module: aggregator (but still without demodata)

2012-07-02

- moved routine `get_module_records` to waslib.php

2012-07-01

- new class SnapshotViewerInline added in preparation for upcoming aggregator

2012-06-22

- [install.php] added the site reply-to: address to the demodata interface
- new theme: Sophia

2012-06-18

- added SIDN to the list of donors

2012-05-31

- various bugfixes:
 - . [main_admin.php] `get_div_parameters()` again correctly shows bullets (bug introduced in 0.90.4)
 - . [waslib.php] `get_friendly_parameter()` now accepts parameter value '0'
 - . [waslib.php] `was_node_url()` correctly now conveys friendly value '0'
 - . [main_file.php] `main_file()` better check on tricks with `../` in path
 - . [filemanager.class.php] `file_url()` better encodes file url
- new module: redirect
- moved two fields (`link_href` and `link_target`) to new module redirect from edit advanced node properties
- moved all translations (`link_href` and `link_target`) from admin.php to redirect.php for all available languages.

2012-05-30

- new module: snapshots

Release 0.90.4 - 2012-04-19 / 2012041900

2012-04-18

- [main_admin.php] selective number of bullets in message list
- bumped year in copyright messages where necessary
- corrected a few typos in translations
- [translatetool.class.php]: standardise on \n as EOL character
- more preparations for upcoming release
- more donors and translators in CREDITS

2012-04-17

- minor tweaks and typos
- translation updates for de, es, fr, pl, tr and zh
- additional languages: ar, da, fa, hu, pt
- more translations for: de, zh
- manifest updated for en and nl language
- preparations for the upcoming new release:
bumped internal version to 2012041900

2012-04-16

- minor adjustments in the skins: we now only display the phrase 'Messages' in the braille skin.
- we now use separate CSS-files for braille and textonly instead of admin_high_visibility.css

2012-04-15

- more support for skins in area manager, translate tool, group manager, file manager, user manager and acl manager.
- more fine tuning in alt-text translations
- skins now also provide toplevel navigation items
- additional style sheet for 'big' skin (PoC)
- additional style sheet for 'lowvision' skin (PoC)
- minor cosmetics

2012-04-14

- we now show the word 'Messages' in the yellow message area to make it easier for the vision impaired to identify error messages
- changed a lot of alt texts to better convey the meaning of icons, ie. what is the effect of clicking on the icon rather than a description of a pretty picture (see <http://webaim.org/techniques/alttext>).

2012-04-13

- typo in translation: not implemented -> not YET implemented.

2012-04-12

- better message stating module manager and statistics are not yet implemented
- [main_admin.php] added display of donor logos to the start centre
- added new theme 'axis'

2012-04-07

- [theme.class.php]: done with Bazaar Style Style Sheets
- [filemanager.class.php]: bugfix: do caseInsensitive check on filename ext.

2012-04-06

- [updatelib.php, tabledefs.php]:
 - . added field 'style' to table 'nodes' (for Bazar Style Style Sheets)
 - . replaced field 'high_visibility' with field 'skin' in table 'users' and updated existing users in database: high_visibility=FALSE -> skin='base'

- and high_visibility=TRUE -> skin='textonly'
- renamed variables \$high_visibility to \$text_only throughout preparing for the introduction of skins
- [usermanager.class.php] relaced field 'high_visibility' with 'skin' in the edit user dialog
- added a 'proof of concept' for the braille skin

2012-04-05

- [about.html]: the logo now links to the project website

2012-03-31

- imported CKEditor 3.6.2 (Minimum setup, ie. without source tree, samples, etc)
- imported the original CKEditor 3.6.2 source tarball in the devel/imports directory
- statement of chosen license for CKEditor added to /program/lib/ckeditor/legal.txt
- updated several files in order to add ckeditor as an editor choice (this includes the install wizard and the demo data)
- started with an update routine to change the acceptable settings for editors in existing installations.

2012-03-07

- [base.css, admin_base.css] minor omissions corrected

2011-10-11

- [rosalina.class.php]: typo in construction of (single) logo

2011-10-06

- [index.php, file.php]: re-arranged check for and redirect to maintenance.html
- [manual.php]: we now offer a choice of manuals if more than one is installed

Release 0.90.3 - 2011-09-30

2011-09-30

- removed obsolete file area.class.php from repository
- removed obsolete file module.class.php from repository
- [updatelib.php]: added two files to list of obsolete files
- [updatelib.php]: renamed the update routine to match the new release version
- last minute update of Spanish: now completely up to date for 0.90.3
- Updated the list of contributors in CREDITS.txt
- Bumped version/release/release date to 2011093000 / 0.90.3 / 2011-09-30

2011-09-29

- updated existing language files for Spanish, Chinese, French
- added new language Polish
- added new language files for German, Turkish
- replace icons for account manager and help button with new versions kindly supplied by Greg Whitaker
- [dialoglib.php]: additional input validation (UTF-8)

2011-09-27

- [demodata.php]: bufix: some demo nodes (siblings) had identical sort orders
- removed obsolete file node.class.php from repository
- removed obsolete file modulelib.php from repository
- [updatelib.php]: we now check for obsolete files too
- minor cosmetic changes

2011-09-26

- [pagemanager.class.php]: get rid of current area if it is no longer there
- [pagemanager.class.php]: bugfix: sort order is now calculated correctly when inserting a new page/section
- [updatelib.php]: added one-time fix for mixed-up sort orders in existing nodes
- New configuration item 'pagemanager_at_end' allows inserting new nodes at the top or appending at the bottom of a (sub)section in the Page Manager.

2011-09-23

- [groupmanager.class.php] re-arranged some code, mainly dealing with group_delete() and group_save() (deletion of group/capacities) and also some cosmetics
- [groupmanager.class.php] added additional warning when user tries to delete her own group/capacity
- [areamanager.class.php]: we now also remove the empty data directory

2011-09-22

- renamed datadir_is_empty() to userdir_is_empty()
- minor cosmetics
- added function userdir_delete() and we now remove the (empty) userdirectory when the user account is deleted
- [groupmanager.class.php]: we now enforce that the files must be removed before the group can be deleted. Also, the user cannot delete a group of which she is a member. Order of delete rearranged to satisfy FK constraints

2011-09-21

- [instal.php] better test on JPG Support/JPEG Support specified by gd_info()
- [waslib.php]: added small routine to test for (user)files in data directory
- [usermanager.class.php]: additional checks before deleting user account, including blocking of delete own account; cosmetics
- [filemanager.class.php]: minor UTF8-related changes + cosmetics
- Global search and replace to change all references to the LOG_XXXX constants (replaced with WLOG_XXX) to circumvent a stupid difference between *nix and win platforms (see task_logview() in toolslib.php for more information).

2011-09-20

- [updatelib.php] Incorporated changes to nodes.modules_id in upgrade routine.
- version.php: bumped internal version from '2011051100' to '2011092100' in preparation for the upcoming release '0.90.3'.
- fixed two warning messages (E_STRICT) by explicitly defining objects \$CFG and \$PERFORMANCE to be of type stdClass (/index.php, /admin.php, /file.php, /cron.php, /program/init.php).
- [areamanager.class.php] fixed error in deletion of area (unknown table 'user_areas', see also 2009-12-03)
- [usermanager.class.php] Streamlined deletion of user account; renamed routine and user records are now deleted in the correct order.

2011-09-19

- [tabledefs.php] Removed the foreign key constraint for nodes.owner_id.
- [tabledefs.php] Removed the foreign key constraint for areas.cuser_id.
- [tabledefs.php] Removed the foreign key constraint for areas.muser_id.
- [htmlpage_tabledefs.php] Removed the foreign key constraint for htmlpages.cuser_id.
- [htmlpage_tabledefs.php] Removed the foreign key constraint for htmlpages.muser_id.
- [sitemap_tabledefs.php] Removed the foreign key constraint for sitemaps.cuser_id.
- [sitemap_tabledefs.php] Removed the foreign key constraint for sitemaps.muser_id.

- [guestbook_tabledefs.php] Removed the foreign key constraint for m_guestbooks.cuser_id.
- [guestbook_tabledefs.php] Removed the foreign key constraint for m_guestbooks.muser_id.
- [mysql_class.php] Improved/streamlined create_table_sql()
- [install.php] and [demodata.php]: adjusted the order of things so we do not violate FK constraints on initial install
- Fixed FK constraint in nodes.module_id by allowing NULL values in case of a section; adapted pagemanager too.

2011-09-16

- [tabledefs.php] Removed the foreign key constraint for nodes.parent_id.

2011-09-09

- Resized several fields in database to stay within MySQL/InnoDB key limits of 767 bytes with charset utf8mb4 [see updatelib.php]:
 - . areas.path: varchar(240) => varchar(60)
 - . config.name: varchar(240) => varchar(80)
 - . groups.groupname: varchar(255) => varchar(60)
 - . groups.path: varchar(240) => varchar(60)
 - . log_messages.remote_addr: varchar(255) => varchar(150)
 - . login_failures.remote_addr: varchar(255) => varchar(150)
 - . modules_properties.name: varchar(240) => varchar(80)
 - . sessions.session_key: varchar(255) => varchar(172)
 - . themes_areas_properties.name: varchar(240) => varchar(80)
 - . themes_properties.name: varchar(240) => varchar(80)
 - . users.path: varchar(240) => varchar(60)
 - . users.username: varchar(255) => varchar(60)
 - . users_properties.name: varchar(240) => varchar(80)
 - . users_properties.section: varchar(240) => varchar(80)
- This applies to both the update routine and the tabledefs for a new installation.

- Adapted dialogdefs in areamanager for new shorter path
- Adapted dialogdefs in groupmanager for new shorter path and name
- Adapted comments in dbsessionlib for new shorter session_key
- Adapted dialogdefs in usermanager for new shorter path and name
- waslib.php: made sanitise_filename() UTF-8 aware

2011-07-14

- main_admin.php: added a special intermediate screen for those intranet-users that accidentally hit admin.php (prevents some confusion)

2011-06-30

- theme.class.php: removed all references to WAS_SCRIPT_NAME: we now use was_node_url() instead.
- theme.class.php: better UTF-8 sanity check in get_address()
- witemap_view.php: better handling of preview_mode

2011-06-29

- waslib.php: many important changes in the way node_id and/or area_id and other paramters is conveyed to index.php; we now accept calls like /was/index.php/35/photo/5/Picture_of_our_field_trip.html but also (still) the equivalent /was/index.php?node=35&photo=5. See get_requested_node() and was_node_url() for more information.
- install.php: improved educated guesses in get_default_install_values()
- waslib.php: slight changes to the was_node_url() code so we can use this routine for areas too
- theme.class.php and rosalina.class.php: changed to use was_node_url() and removed a few superfluous routines (Theme::friendly_bookmark() and ThemeRosalina::rosalina_href()).

2011-06-27

- install.php: we now use trick with an invisible img to determine whether we can use the friendly URL feature
- added a few more keywords to the demo data of frugal theme demodata

2011-06-25

- install.php and init.php: added complex routines to make sure that we use the correct value of the currently executing script. See the (long) comment for install_script_name() in install.php
- waslib.php: streamlined internal url handling (was_url(), was_file_url())

2011-06-19

- minor additions to install wizard: better processing of demo data

2011-06-17

- refined context-sensitive help in tools menu

2011-06-09

- minor tweaks in schoolyard theme

2011-06-07

- theme.class.php: added html-class 'current' to LI-tag in navigation bar too
- theme.class.php: extra html-classes 'activepage' and 'activesection' in menu
- added theme 'schoolyard' (designed by David Prousch, May 2006) to Website@School

2011-06-03

- installer: make demo data more accessible for additional themes and modules; we now also keep track of nodes, groups and users
- added theme 'rosalina' (based on HV Menu by Ger Versluis (<http://www.burmees.nl/>) to Website@School

2011-05-31

- pagemanager.class.php: fixed a bug in calculate_updated_sort_order(): the calculations went wrong when sorting within a subsection

2011-05-29

- corrected a stupid typo in demodata.php
- streamlined the fetching of module_id's in demodata.php
- theme.class.php: show area in breadcrumbtrail for quicklinks too

2011-05-28

- pagemanager.class.php: better error handling in connecting and disconnecting modules and nodes
- sitemap module:
 - . added an optional header and introductory text to sitemap configuration
 - . various cosmetic changes
- added a real sitemap to the demodata, including a non-blank header and introduction (in all languages)

2011-05-27

- waslib.php: renamed function build_tree() to tree_build and changed the function calls everywhere else (theme.class.php, pagemanager.class.php, aclmanager.class.php, main_index.php)
- waslib.php: moved routine calc_tree_visibility() from theme.class.php because it is even useful outside the theme
- renamed calc_tree_visibility() to tree_visibility() and changed calls
- added initial version of the sitemap module

2011-05-26

- theme.class.php: various cosmetics regarding the separators between quicklinks at the top/bottom of the page

2011-05-20

- waslib.php: added a routine was_url() to qualify relative URLs
- theme.class.php:
 - . the [Go] button is now always visible (reverses earlier decision to embed it in noscript-tag)
 - . autosubmit now works with this.form.submit() instead of by naming the form
 - . added class variables for separators in quicktop, quickbottom, breadcrumb trail (easier to change)
 - . we can now indicate which menu to show via additional parameter menu_id in get_menu()
 - . code cleanup / increased readability
 - . bugfix: we now actually obey the global 'friendly_url' configuration parameter
 - . modified friendly_bookmark() to deal with UTF-8 and diacriticals in a slightly less ASCII-centric way

2011-05-18

- theme.class.php: we now add the area name to the breadcrumb trail
- theme.class.php:
 - . prepare the jump menu in the constructor (saves a trip to the database); adds an additional variable \$jumps to the class
 - . minor cosmetics

2011-05-17

- theme.class.php: the [Go]-button in the area jump-menu now only appears when the user has switched JavaScript 'off', otherwise we autosubmit the form.

Release 0.90.2 - 2011-05-11

2011-05-11

- configassistant.class.php: cosmetics in formatting of float parameter
- theme.class.php: always indicate UTF-8 via http-equiv too
- theme.class.php: minor cosmetics
- version.php: bump release and release date to 0.90.2 and 2011-05-11 (the internal version was already bumped two days ago)

2011-05-09

- init.php: we now always include utf8lib.php with essential UTF-8 utilities
- mysql.class.php: corrected stupid typo in dump().
- useraccount.class.php: yet another stupid typo (in Useraccount())
- updatelib.php: database conversion from charset 'whatever' and collation 'whatever_whatever_ci' to 'utf8' and 'utf8_unicode_ci' (or 'utf8mb4' and 'utf8mb4_unicode_ci').
- version.php: bumped internal version from '2011020100' to '2011051100' in preparation for the upcoming release '0.90.2'.

2011-05-08

- mysql.class.php:
 - . cosmetic/documentation changes
 - . added code for full / partial utf8 support (partial <5.5.3, full 5.5.3+)

- . set charset to utf8 or utf8mb4 on connect()ing to the database
- . also remember the MySQL-version in the backup (dump())

2011-05-06

- install.php: https on a non-standard port yielded the wrong port number
- utf8lib.php: added a function to map characters with diacriticals to plain ASCII as far as possible
- install.php: added the utf8_strtoascii() mapping when generating data directories; makes it more readable for some languages compared to simply deleting all multibyte UTF-8 characters
- demodata.php: better UFT-8 support and better sanitised directory names

2011-05-05

- added Chinese translation (based on 0.90.1)
- updated Spanish translation: now also based on 0.90.1

2011-05-04

- loginlib.php: explicitly go to 'index.php' in 'home' link in login dialog
- install: added some more comments for translators in Ennglish languages files
- added French translation (based on 0.90.1)
- we now refer interested parties to <http://websiteatschool.eu> for a full overview of credits (see CREDITS.txt).

2011-05-02

- install.php:
 - . suppress PHP-errors when realpath, opendir are outside the open_basedir tree
 - . more checks on possible UTF-8 issues
 - . added warning for obsolete MySQL-version (< 4.1) in database and compatibility screens (including Dutch and Spanish translation (partial))
- filenmanager: added the untranslated path to directory overview in task_list_directory()
- dialoglib.php:
 - . slight change of the meaning of ~ as hotkey-indicator: we now only accept ASCII-characters. UTF-8 characters are silently ignored.
- main_admin.php: we now go explicitly to 'index.php' when navigating to the pulic area so a bare CFG->www_short will not take us to say index.html or default.htm.
- pagemanager.class.php: same for the preview function: explicit link to index.php.

2011-04-29

- install.php:
 - . UTF-8 support for password validation routine
 - . UTF-8 aware use of substr()
- uitf8lib.php:
 - . added function utf8_substr()

2011-04-22

- install.php:
 - . more support for UTF-8 in save_database()
 - . additional input checks on absurd lengths

2011-04-18

- install.php: all input (GET/POST) is now checked for UTF-8 conformance

2011-04-15

- created utf8lib.php with essential routines for handling UTF-8 (based on Unicode 6.0.0, February 2011)
- install.php:
 - . we now explicitly check user input for valid UTF-8
 - . added 'accept-charset="UTF-8"' to all forms (perhaps superfluous)
 - . improved caseInsensitive check for the 'I Agree' prompt

2011-04-13

- install.php: added http-equiv Content-Type: text/html; charset=UTF-8

2011-03-11

- bugfix: there was a problem with the database dump: NULL-values were not properly dumped.
- clarification: all relevant fields in the database can accomodate an IP-address of 15 or 39 characters because the relevant fields are varchar(255)
- finetuning the message-id header field in email class
- updated INSTALL.txt with instructions for upgrading and installing the manual
- replaced email address for submission of translations (in the translate tool)

Release 0.90.1 - 2011-03-09

2011-03-09

- bugfix: a newly added language in the translate tool was always labeled 'active' even when the box was unchecked; fixed in translateTool.class.php
- re-arranged the status overview table in updatelib.php: we now have additional columns for release date and release.
- a new language without a corresponding manifest but added/created locally can now exist in the database without raising an error in the update status overview
- changes in /program/version.php:
 - . Bumped WAS_RELEASE from 0.90.0 to 0.90.1
 - . Bumped WAS_RELEASE_DATE from 2011-02-01 to 2011-03-09
 - . WAS_VERSION remains unchanged at 2011020100
- bumped release/release date in various manifests too (languages, modules, themes).

2011-03-08

- done with install_theme() and install_module(): we can now install new themes and modules
- corrected a few typos
- added the Spanish translation of v0.90.0 to CVS
- bumped all versions and releases of all existing languages, themes and modules to 2011020100 / 0.90.90 / 2011-02-01 in order to make it clear to which core version a particular subsystem belongs. Old (internal) versions: frugal: 2008022000, htmlpage: 2008112500, en: 2010092700, nl: 2010092700. This requires a manual update in the update manager (which is in fact a no-op).

2011-03-07

- added more clarification in `htmlpage_install.php` about inner workings
- corrected a few typos
- modified a few translations in 'en' and 'nl'
- added the routine for updating existing modules
- started on `install_theme()` and `install_module()`

2011-03-03

- added some more clarification in `frugal_install.php` about the inner workings of the call-back routines
- we can now call the `theme_upgrade()` call-back from the Update Manager (`updatelib.php`).

2011-03-02

- streamlined the update status overview in `updatelib.php`
- added code to update and install additional language packs in `updatelib.php`.

2011-02-21

- updated version of `manual.php`

2011-02-18

- bugfix: typo in `/program/manual.php`

2011-02-03

- the new Project Home Page is now at <http://websiteatschool.eu>; many, many sourcefiles modified to include the new location (mainly in copyright messages)

Release 0.90.0 - 2011-02-01

2011-02-01

Today the complete development of WebsiteAtSchool was transferred to CVS at the development site BerliOS (<http://developer.berlios.de>), using the current version (which we used to call 0.1.7).

We now start with a new phase in the development, i.e. a publicly visible CVS-server and official releases etc. This is a good opportunity to bump both the internal version and the external version (release) and start with a more or less clean slate.

Furthermore, we will be generating a first 'official' release today.

Changes in `/program/version.php`:

- Bumped `WAS_RELEASE` from 0.1.7 to 0.90.0
- Bumped `WAS_VERSION` from 2010122100 to 2011020100
- Bumped `WAS_RELEASE_DATE` from 2010-01-20 to 2011-02-01

Changes in `/program/lib/updatelib.php`:

- Added logic to update existing database versions to 2011020100

```
*****
*****
*****
*****  BELOW THIS LINE ONLY OLD AND OBSOLETE CHANGES  *****
*****
*****
*****
*****
```

2011-02-01

- changed development colours to something more appropriate

2011-01-20

- completely overhauled version of devel/tools/makedist.sh: we can now generate 'official' releases too
- updated copyright/license message in scripts in devel/tools
- bumped release from 0.1.6 to 0.1.7 just to test the release procedure. After checking in version.php we need to add a revision tag with 'cvs rtag release-0_1_7 was' and see what happens.

2011-01-12

- added a simple counter/index to translate tool: makes it easier to refer to a particular translate string on the phone
- added a devel tool to identify changed and added strings between versions in a convenient single HTML-page (langdiff.sh in the tools directory)
- corrected a stupid typo in English and Dutch translations
- bumped version from 0.1.5 to 0.1.6

2011-01-10

- Changed bottom line in main_admin.php: we now only display the execution time, # of queries and TOD when in debug-mode.
- Removed the (empty demo-like) stubs in the group manager and user manager indicating configurable modules per user/group. This makes it easier to create realistic screenshots for the upcoming manual.
- Bumped release from 0.1.4 to 0.1.5

2010-12-20

- completed the automatic execution of the update manager (called whenever there is a mismatch in the core version)
- added update manager to tools menu
- bugfix: removed double escape in version check icon in start center
- bumped release from 0.1.3 to 0.1.4
- bumped version from 2010120800 to 20101221 (just for testing purposes)

2010-12-17

- started with support for automatic update wizard (driven by internal version number mismatch)
- bumped version to 0.1.3

2010-12-13

- bugfix: missing closing '>' in buttons in install.php
- bumped release to 0.1.2
- bugfix: error with deleting of sections/nodes in pagemanager.

2010-12-10

- bugfix: if CFG->www_short is empty, the link to the public site was broken [main_admin.php]
- bumped release to 0.1.1

2010-12-08

- mass update of all source files, now with the correct copyright text and link to license.html etc. (continued)
- renamed DatabaseResult to DatabaseMysqlResult to prevent name clashes once DatabasePostgresql and DatabasPostgresqlResult once these are written

- cleaned up various files (removed obsolete chunks of dead code)
- added the 'registered trademark' message to the loginlib
- removed the old (wrong) logo from the standard theme
- updated base.css to increase height of quickbottom div
- added a rel link to /program/graphics/favicon.ico (in main_admin.php)
- bumped external version from 0.0.7 to 0.1.0
- bumped internal version from 2010092700 to 2010120800

2010-12-07

- cleanup of init.php; streamline the error_exit() routine
- mass update of all source files, now with the correct copyright text and link to license.html etc.
- various cosmetics
- added link to on-line license.html in LICENSE.txt too

2010-12-02

- added the final version of license.html with all the legalese
- added /program/about.html with 'Appropriate Legal Notices'
- modified the installer to check for the exact version of license.html
- added global definition: WAS_ORIGINAL in version.php
- added some more logic to the online version check
- bumped release from 0.0.6 to 0.0.7
- added a few graphics files for 'powered by' and 'based on'
- added a new logo file of 284x71 (instead of 248x53)
- incorporated the 'powered by' and 'based on' graphics in install.php
- incorporated new logo in install.php
- header height in base_admin.css from 53 to 71 pixels for new logo
- added link to about.html via appropriate_legal_notices() in install.php
- added link to about.html via appropriate_legal_notices() in main_admin.php
- modified the translatable text in footer: we now force the English 'powered by' (or 'based on') even in other languages.
- added an explanation for creating modified versions in '/program/about.html'
- also changed the address for new users from schoutdi@knoware.nl to online@websiteatschool.org

2010-10-28

- raised the priority of logmessage when a virus is detected in filemanager.virusscan()

2010-10-27

- we now use UTF-8 in the default Theme class; send it to browser via http-headers
- stupid typo in Theme class corrected (bugfix)
- new function: log view (in tools menu)
- added two more job permissions and changed existing: we now have separate permissions for translate tool, backup an logview.

2010-10-26

- minor cosmetics: we now have a database backup named host-database-prefix-date-time (either .zip or .sql)
- we now have a 'hidden' feature that allows for downloading the uncompressed .sql by specifying 'download=sql' instead of 'download=zip'.

2010-10-21

- bugfix in rfc2047_qstring(): we now keep UTF8-tails together

- with the first octet in the sequence in the same 'encoded-word'
- completed backup function in tools menu

2010-10-20

- implemented a (database-specific) database/SQL-dump in mysql class
- first draft of database dump downloader in tools (no check on user perms yet)
- minor cosmetics

2010-10-19

- moved a few mail-related routines from waslib to email.class
- adapted loginlib: we now use email class to send mail
- added a header implying that we send UTF-8 in login dialog too
- added an extra download feature: file.php/websiteatschool/languages now returns a ZIP-file with all user-defined translations of active languages

2010-10-18

- added a supersimple mailer class in email.class.php
- adapted translatetool to use this new email class
- minor cosmetics

2010-10-17

- fixed some bugs in filemanager.class.php
- preliminary changes in interfacing to mail() command (viruscan)
- cosmetics

2010-10-15

- implementation of RFC2047-style quoted-printable mail header extensions in translate tool

2010-10-11

- implementation of conversion to quoted printable according to RFC2045 section 6.7 (waslib.php)
- adapted routine to submit translations to deal with quoted printable message
- (finally) converted the program's native language to UTF-8, at least in AdminOutput in main_admin.php (we still need to deal with UTF-8 in the database)

2010-10-06

- translatetool.class.php: we can now really save user translation files under CFG->datadir/languages/ (submit of translations is still work in progress)
- (later) initial version of sending translation works (more or less).
- added a cache reset to \$LANGUAGE to force re-read of the userfile just written

2010-10-05

- minor changes in language.class.php:
 - . we now cache a list of _all_ (active+inactive) languages
 - . removed obsolete and unused routine get_languages()
 - . renamed get_language_names() to get_active_language_names()
- minor changes in files which relied on the old language.class.php
- nasty bug: a '.' in a fieldname yields a '_' in \$_POST. Huh?
workaround: use a ':'

2010-10-04

- translatetool:

- . we now have visible codes in the translation dialog.
- . we also have fields for metadata (per full_domain)
- . still work in progress

2010-09-30

- translatetool: we can now show a domain menu in the edit translation dialog but otherwise it is still work in progress

2010-09-29

- translatetool: we can now add new languages and edit existing ones but it still work in progress

2010-09-28

- translatetool is work in progress

2010-09-27

- changed the way languages are handled: we now store the name of the language expressed in the language itself in the languages table which required a change in the tabledefs
- bumped version from 2010070700 to 2010092700 because of the changes in the database (table definition of 'languages').

2010-09-23

(after the summer break)

- bugfix (finally) in language class where valid translations were overwritten in memory when another phrase key was not found.

2010-07-08

- we now have a distinction between browsing files limited by filename extension and uploading files limited by extension. bottom line: in the file/image/flash browser the user can only see and upload selected files, in the file _manager_ the user can upload selected files but see _all_ files, including 'forbidden' files (eg. some random script uploaded by an intruder)
- new feature: if a file is detected (in filemanager) which would currently not pass the test for allowable upload, the entry in the filemanager listing is displayed with the CSS error class and the preview links are completely disabled, preventing the user from accidentally previewing a rogue file.

2010-07-07

- now suppress menu too in filebrowser/imagebrowser (via .CSS)
- started with support for new job 'flashbrowser' (cousin of 'imagebrowser')
- added support for three new configuration parameters that limit the allowable file extensions (as a shorthand for allowable file types)
- bumped version from 2010063000 to 2010070700 because of the three new configuration options in the installer

2010-07-02

- initial version of thumbnail-based file/image browser (still incomplete)

2010-07-01

- added check on GD in installer
- added a hint to print the summary page in installer

2010-06-30

- bumped WAS_VERSION from 2010052400 to 2010063000 because of a typo last week (it should have read 2010062400 but the actual

value was 2010052400). Also, the new default value for 'thumbnail_dimension' decreased from 150 to 100 pixels.

- added support for thumbnails in the basic style sheet (based on the new default dimension of 100).

2010-06-24

- added generation of thumbnails to file upload routine in file manager
- added more about limits (file size, upload size) in file upload explanation
- minor cosmetics
- bumped WAS_VERSION from 2010051300 to 2010062400 because of new configuration option 'thumbnail_dimension'

2010-06-22

- re-arranged code and added new file /program/lib/filelib.php

2010-06-15

- filemanager extended so it can double as a filebrowser via job=filebrowser and image browser via job=imagebrowser
- linked FCKEditor to the filebrowser and imagebrowser

2010-06-14

- draft version of link browser combined in filemanager (filebrowser)

2010-05-04

- done with file upload except for 1 small thing (sanitise file type)
- minor cosmetics here and there

2010-05-13

- renamed new parameter 'files_upload_count' to 'upload_max_files'
- bumped WAS_VERSION from 2010051200 to 2010051300 because of this
- minor cosmetics here and there
- implemented F_FILE in dialoglib

2010-05-12

- added comptibility check for Safe Mode (should be 'Off') in install.php
- added support for clamscan anti-virus in install.php
- added extra configuration options in site config for clamscan
- added configuration option fo maximum number of simultaneous uploads
- bumped WAS_VERSION from 2010010700 to 20100512 because of extra site configuration options

2010-04-14

- almost done with filemanager; what remains is file upload implementation

2010-04-13

- added file delete confirmation dialog to filemanager

2010-03-31

- filemanager: added a Javascript one-liner to select all files
- filemanager: we can now add (create) subdirectories

2010-02-13

- filemanager:
 - . directory listing layout done
 - . sort option for three columns (ascending or descending)
 - . human-readable filesize
 - . various stubs for delete and add file/dir

still work in progress though

2010-02-10

- filemanager: navigation basically works but still: work in progress

2010-02-08

- filemanager is still work in progress

2010-02-05

- added some more navigation to file manager

2010-02-03

- traded filemanagerlib.php for filemanager.class.php

2010-01-28

- for now done with main_file.php: we can serve files from the data directory using the correct (?) mime types etc.

2010-01-20

- Oops. Bug in tabledefs: there was no unique index on areas.path. Fixed.
- Done with download of source code: we now recognise file.php/websiteatschool/program (program code) and file.php/websiteatschool/manual (manual in current language)
- More oops. Another bug in tabledefs: we didn't have the unique index on path for users and groups either. Fixed.
- Busy with file.php: validation and access control done.

2010-01-19

- started on file.php and download of source in .ZIP-file

2010-01-13

- created a new class Zip (in /program/lib/zip.class.php) which allows for creating ZIP-archives on the fly, including Deflate compression, file and archive type comments and zipping from memory or from existing file.

2010-01-09

- removed a lot of superfluous files from /program/lib/fckeditor
- suppressed the option 'fcksource=true' in dialoglib: the user that sets that parameter in \$_GET gets the plain text area instead of the source-variation of FCKEditor (which is mostly deleted from /program/lib/fckeditor and wouldn't work anyway).

2010-01-08

- added the full tarball FCKEditor_2.6.5.tar.gz to devel/imports so we have the original source nearby
- added type F_RICHTEXT as a gateway to the FCKEditor in dialoglib
- first steps to actually incorporate FCKEditor 2.6.5 in directory /program/lib/fckeditor ia CVS import function. We consider the extensions gif png swf pfx and fla to be binary files, the rest done via -ko.

2010-01-07

- added view-only field in edit user that shows the data directory name
- removed the 'advanced' option from the edit user menu because there is no 'advanced' dialog (yet)
- added editor selection to basic user properties, choices are:

plain and fckeditor

- added the logic for groups path parameter
- added r/o view of data dir name in edit function in areamanager
- added r/o view of data dir name in edit function in usermanager
- added r/o view of data dir name in edit function in groupmanager
- corrected some typos
- added unique index on path field in areas table
- bumped version from 2010010600 to 2010010700 because of changed tabledefinition
- added the data path to the add area dialog
- update demodata so we have readable area data dirs now instead of numbers

2010-01-06

- we now also make subdirectories 'users' and 'groups' in the data directory (next to 'areas' and 'languages') in the installer
- we no longer record the words 'areas' and 'users' and 'groups' in the path fields in the corresponding tables, dirs are strictly separated now.
- added global configuration option for generating friendly URLs instead of querystrings
- added routine to sanitise filenames to waslib (used in creating datadirectories for areas, users and groups)
- we now check to see if the user tries to install with a username that is also used in the demodata
- we now actually create the datastructures in the datadir for all demo-data users, groups and also the areas
- bumped version from 2009120300 to 2010010600 because of changed tabledefinitions
- added more translation texts for new config items 'friendly_url' and 'editor'
- fixed a few typos

2009-12-22

- done with pagemanager class
- removed obsoleted pagemanagerlib.php

2009-12-21

- almost done with pagemanager class

2009-12-19

- 'playing' with stylesheet: we now see line-through on 'dimmed' links in high-visibility mode (works for funnel mode)

2009-12-18

- added funnel mode support to AdminOut

2009-12-17

- very busy re-arranging code in page manager

2009-12-08

- changed the has_...permissions() routines in useraccount: we now test for 1 or more permissions, not an exact match anymore: (\$perm & \$mask) != 0 versus (\$perm & \$mask) == \$mask.

2009-12-04

- finally decided on the issue of 'per-area' datadirectories: we simply use \$CFG->datadir/areas/<area_id>

- reworked the permissions in the area manager
- we no longer allow changes in the path property of areas; the field isn't even shown anymore.

2009-12-03

- finally got rid of the tables `users_areas` and `users_nodes` because they are now replaced by the `acls-tables`.
- removed fields `job_permissions` and `permissions` from `users` table because they are now replaced by the `acls-tables`.
- implementing `acls` in the `useraccount` class (work in progress)
- bumped version from 2009102100 to 2009120300 because of changed/removed tables
- removed `PERMISSION_VIEW` from `useraccount.class.php` because it is now replaced by `ACL_ROLE_INTRANET_ACCESS`
- removed `PERMISSION_ADMIN` (which was added 2008-10-01) from area manager. It still needs to go from `pagemanager`.

2009-11-30

- added an item to the CMS dialog in the installer to query the user for a generic demonstration data password (will be assigned to all demo accounts)

2009-11-04

- done with `demodata`: we now have a public area with 16 pages and 6 sections and a private area with 8 pages and 3 sections, alltogether in 2 languages (en and nl). Pfew!

2009-11-03

- `demodata()` now adds 3 areas, 4 groups and 8 users in 2 languages
- `demodata()` now also adds many nodes in 2 languages (work in progress)
- reduced the `htmlpage_demodata()` and `frugal_demodata()` to no-ops because all `demodata` is already inserted in the main `demodata()` routine.

2009-11-02

- finished logic for `datadirectory` name based on `dataroot` and quasi-random subdirectory (to obfuscate the `datadir` if it happens to be located within the document root)
- we now create an empty `index.html` in both `dataroot` and `datadir` to prevent leakin information if `dataroot` is located within document root.
- added creation of subtree 'languages' to `datadirectory`, also with empty `index.html` files

2009-10-23

- re-arranged the interface via manifest-files: we now fill the array `manifests[]` (plural), just like `tabledefs[]`
- main logic for installing is done, only thing left to do is insert `demodata`
- quick fix in set defaults for `cms_dir`

2009-10-21

- yet another changed `tabledef`: bumped version from 2009102000 to 2009102100
- we now install themes and languages too (not just modules)

2009-10-20

- bumped version to from 2009101600 to 2009102000 because

- of changed tabledefs 'modules', 'themes' and 'languages'
- added manifest files for languages 'en' and 'nl'

2009-10-19

- changed manual.php: we made a start with linking topics and subtopics to html-files under /program/{language}/
- changed handling of help_topic in install wizard (subtopics)
- we now actually create the data directory if it doesn't exist already
- we now return a link to check for new versions rather than the simple assumption 'OK' in the compatibility dialog in the install wizard

2009-10-16

- removed fields manifest_script and install_script from modules table; if necessary we can construct these from the manifest or use 'well known' script names (like 'tabledefs.php' and {\$module}_manifest).
- Bumped version from 2009061100 to 2009101600 because of change in the datadefinition of the modules table.
- We now are able to install modules via their manifest (but not yet demodata)

2009-10-15

- extra debugging in database library and mysql class
- almost done with actual installer

2009-10-13

- added more documentation to functions
- added functionality to write config.php (if filesystem permits)

2009-10-12

- added a check on already installed
- added optional version check in compatibility screen

2009-10-05

- the installer is still work in progress but all the dialogs are done

2009-10-01

- more checks/validation in the installer

2009-09-30

- re-arranged the code in install.php: we now have the InstallWizard class
- re-arranged the order of the funnel: we now require valid database credentials before we leak any information
- added delays to the database validation if something goes wrong, to scare off the spooks

2009-09-29

- added suppress_output mode to AdminOutput class

2009-09-24

- installer: work in progress

2009-09-23

- installer: work in progress

2009-09-22

- installer: work in progress

- added an ad-hoc image for an OK-button

2009-09-21

- installer: work in progress

2009-09-18

- added style for previous and next buttons to admin_base.css + icon

2009-09-10

- we now require a new password twice when adding a new user
- code cosmetics

2009-07-24

- done with pagemanager permissions in usermanager and groupmanager; we can now store and retrieve permissions on node level.
- added minor cosmetics in the expanding/collapsing of areas in aclmanager: we now always keep the section to expand/collapse visible on the current screen by setting the offset to the offset of that area
- updated translations

2009-07-23

- after a lot of time I finally got the pagemanager permissions in the ACL Manager right: we can now actually save the settings via a complex dialog (for the programmer) that deals with screens (in long lists) and also allows for opening and closing areas.
pfew

2009-07-01

- more work on the pagemanager permissions dialog, including walking the node tree in an area (work in progress)

2009-06-30

- moved pagination code to AdminOutput()

2009-06-19

- we now have fully implemented the 'related-acls' feature for the user intranet access dialog and the user admin access dialog.
- some more streamling in the install.php to navigate in tabledefs and datadump

2009-06-18

- we now also delete user associations with a group/capacity when a capacity is removed from a group in the groupmanager
- (special request) we now have a jump-menu in the theme via which a visitor can jump to any area available to her (logged-in users also see private areas if they have permission)
- added a users+groups summary to the account manager intro screen
- added support for user privileges intranet and also user privileges for admin, the latter including the related feature
- various typo's corrected

2009-06-17

- we can now manipulate a user's group memberships: view, add, delete

2009-06-16

- completed pagination routine with a limit on links to show
- changed default number of links to show to 7 (was 5)

2009-06-11

- added pagination to users list in user manager
- added configuration parameters for pagination to \$CFG via config table
- added 'memory' to users list: after editing you return to list/screen you started from (if possible)
- bump version to 2009061100

2009-06-05

- added function for sorted list of languages to \$LANGUAGE
- saving basic properties of user now works

2009-06-04

- minor tweaks in loginlib in order to reuse the password/salt-code
- added 'add user'
- added 'delete user'
- started with 'edit user'
- added more Dutch translations, again

2009-05-31

- done with acl for admin jobs
- fine-tune in groupcapacity menu: suppress pagemanager link if this group/capacity has no permission for page manager
- more or less done with users overview (but Dutch translations lack)

2009-05-30

- code cleanup in group manager
- more logging in group manager
- some small convenient enhancements to the install.php making it easier to keep track of the logging
- delete group function now works
- added more Dutch translations

2009-05-29

- we can now save group data, including checks on ACLs etc.

2009-05-28

- busy with implementing ACL-configuration starting with intranet permissions
- added more and more information to the demodata: now with an acls-record for every user and every group/capacity.

2009-05-26

- typo in tabledefs; bumped version to 2009052600

2009-05-22

- added a breadcrumb trail to the admin output object because we really need this to remember where we are in the account manager
- added list of users per group-capacity (task=capacityoverview)
- minor cosmetics (including correction of type in name Helen Parkhurst)
- busy with the navigation between various screens dealing with group-capacity properties

2009-05-20

- added some 55 dummy-users and 15 dummy-groups to demodata in order to 'play' with the account manager
- we now are able to add new groups, including up to 8 capacities per group

2009-05-19

- group manager: we can now show a list of existing groups and group-capacities including clickable links
- added some group data to the demodata
- bumped the version to 2009051900 to make sure that the demodata is re-created in the testversion

2009-05-18

- updated logo's, now with ®
- added new icon for account manager
- moved usermanager around together with the group manager; there is now a top-level accountmanager for users and groups

2009-05-14

(after some two months of thinking we're back with more)

- start with better access control via 6 new tables
- added three tables for additional group-based access control
- obsoleted two tables (users_areas and users_nodes) and some fields in the users table: the functionality is now available via ACLs.
- bumped versions to 0.0.5 and 2009041500
- Deprecated the 'unsigned' attribute in tabledefs because this is a non-standard feature which MySQL happens to implement but other DBMSs might not. In general 2^{31} integers should suffice anyway.

2009-03-18

- rearranged code for calculating the URL for version checker on project's website
- added support for Bazaar Style Style Sheets on static level and on area level (not yet on node level; todo)

2009-03-17

- fixed bug in dialog_validate(): viewonly fields should not need to be validated because they are not supposed to change anyway
- added routine for manipulating/editing the main site configuration (via ConfigAssistant)
- got rid of get_configuration(): we now use get_properties() for the main config too
- done with the area manager except 1 little detail (the handling of the data directory)
- added support for datatypes 'l' (picklist) and 'r' (radio) in ConfigAssistant; the type 'c' (checklist) still needs to be done

2009-03-14

- bumped version internal number to 2009031400 because of significant changes in the database (the extended configuration table format)
- added begin of support for 'Cascading Cascading Style Sheets' or 'Bazaar Style Style Sheets'.
- added yet more demodata to show the possibilities of the area manager

2009-03-06

- almost done with area manager; we only need to copy the theme parameters when adding a new area
- rearranged/renamed some phrases in order to keep at least a little overview

2009-03-05

- added a configassistant class as a quick tool for editing generic configuration data (first used in editing theme properties in areamanager)

2009-02-27

- moved all code dealing with area management to seperate class

2009-02-26

- added functionality to delete areas
- started with functionality to add areas

2009-02-25

- added fields mtime, muser_id, ctime, cuser_id to areas table
- bumped internal version to 2009022500
- bumped external version to 0.0.4
- added copy of webmaster account to account 'textmaster' to demodata: easy checking text interface with permissions of webmaster
- optimising queries on 'nodes' table by adding an index on field 'area_id'

2009-02-24

- (finally) started with configuration manager
- added has_site_permissions() to user class so we can use access control in area manager
- moved some code from pagemanagerlib to waslib so we can reuse it

2008-12-14

- incorporated a stub with base.css so we can see what a page will look like (more or less)
- minor cosmetics in the demo data

2008-12-10

- more navigation in base theme: now tree-like menu
- added language files en and nl to frugal theme
- minor cosmetics

2008-12-08

- added table themes_areas_properties
- added more demo data, including a quicklinks top + bottom in hidden sections
- changes in the built-in theme object
- bumped internal version to 2008120800
- added some more functionality to the built-in theme object

2008-12-04

- removed global \$NODE and \$AREA; we use a local variable \$area in main_index() and information about the node is to be found in the local variable \$tree
- obsoleted file area.class.php
- obsoleted file modulelib.php
- obsoleted file module.class.php

2008-12-03

- moved some more code from pagemanagerlib.php to waslib.php
- we no 'fix' circular references in function is_under_embargo() (this should be fixed and properly done in a database repair tool or something)

2008-12-01

- moved build_tree() to waslib because it is used from main_index() too and not just from main_admin()

2008-11-26

- some files re-arranged and renamed: e.g. /program/lib/adminlib.php now lives in /program/main_admin.php
- added stubs to /cron.php (which call /program/main_cron.php) and /index.php (which call /program/main_index.php)
- removed adminlib.php from CVS repository because it is now obsolete
- added begin of support for Apache's PATH_INFO feature for specifying area and node like index.php/aaa/nnn or index.php/nnn. This does pose a few problems with relative urls... to be fixed

2008-11-25

- yet another change to the database: modules table
- version bumped to 2008112500

2008-11-24

- added more logic to the garbage collection and expiry of obsolete sessions
- added unconditional garbage collection after successful login
- added a default value for session_expiry to demodata
- renamed global config option sessionname to session_name throughout
- version bumped to 2008112400 because of change in demodata

2008-11-21

- first attempt to actually save htmlpage content
- version bumped to 20081121 because of change in data definition

2008-11-20

- added three stubs for modules: htmlpage, mypage and guestbook
- we now actually delegate the task of editing content to the module admin code

2008-11-17

- we now take readonly into account when deleting nodes
- documentation in dialoglib
- minor cosmetics

2008-11-10

- save advanced node properties now working (except moving a non-empty subtree to another area)
- bumped internal version number (WAS_VERSION) from 2008020100 to 2008111000. This forces an error message/condition code 050, which means that the database is not in sync with the program. Normally, this would mean that an updater should be executed which changes the database structure. In the development phase we simply need to reload the demodata. Naturally I changed the version number in the demodata too.
- I also bumped the external version number from 0.0.1 to 0.0.2, just to gain some experience with changing versions.
- Added a FAQ about condition code 050.
- Added a field 'locked_since' to the nodes table, allowing to make the 'sorry, record is locked' message even more friendly

2008-11-09

- minor changes:
 - . default visibility for a new page is now 'hidden' as per Dirk's request
 - . field order in basic node properties now has module as last in the list

- a 'default node' now loses the default node bit when the node is moved to another parent (preventing ending up with two defaults).
- added a class 'current' to options in the menu (currently there are two menus: the area menu and the node edit menu). The current item is underlined (via CSS).

2008-11-08

- we now manipulate the sort order in such a way that the sort order of other nodes is increased/decreased in order to make room for the moved node.

2008-11-07

- added stubs for `module_connect()` and `module_disconnect()` so we can actually link a module to a node (and unlink too, obviously)
- we can now almost save the data from the basic properties dialog (everything except the sort order by itself)

2008-11-06

- we can now fill the dialogs for editing a node with data from database
- added Dutch translations for all prompts etc.

2008-11-05

- rearranging the translations in `/program/languages/em/admin.php`
- minor change in `admin_basic.css` so we can indent options in a drop down list (tree)
- added a comment feature to the English translation file `admin.php`: this allows for communicating the purpose and context of strings to the translators
- completed the dialogs for basic and advanced editing of nodes (but not yet the actual saving etc.)

2008-11-04

- removed `PERMISSION_NODE_EDIT_CHILD` from `useraccount` class, changed into `PERMISSION_NODE_EDIT_NODE` because that makes more sense, I think.
- started with edit a node dialogs
- cosmetics

2008-10-31

- cleaning up code (it takes time to condense rough code and make it more elegant)
- added better logging to the alert feature
- on special request: now new nodes are added at the top of their section rather than at the end (some logic is more logic than other logic)
- we play a special trick when adding a node: we force switch to custom view with all ancestors opened and any other sections closed.

2008-10-30

- added a routine that churns out alert messages every now and then
- added more logging + alert to change default routine
- minor cosmetics

2008-10-29

- we now can add new sections and pages
- we now accumulate alerts when a new page/section is added

2008-10-28

- a small diversion in the past week: add tables for alert function (alerts, alerts_areas_nodes)
- updated demodata to 'play' with alerts
- added function to lock records of nodes table, alerts table

2008-10-20

- added even more functionality to dialoglib.php + documentation
- first attempt to do a dialog in pagemanager (no save yet) with new lib

2008-10-18

- added more functionality to dialoglib.php

2008-10-15

- added dialoglib.php

2008-10-13

- moved some html-utilities to new file /program/lib/htmllib.php
- added some links + translations to the start center
- first start on 'add a section' code

2008-10-10

- re-arranged icons, added new icons for visible/invisible page
- modified page preview to use a one-time access code via md5-hash
- documented all routines in pagemanagerlib.php, code cleanup
- implemented the 'set home' function for nodes

2008-10-09

- changed layout of node tree a little bit:
 - . add a node is now displayed at the top of the tree
 - . we no longer show sort_order and node_id in the anchor text (we do show the node_id in the mouse-over (title) though)
- added logic to switch between collapsed, customised, expanded tree view
- we now have stubs in place for all tasks in pagemanager, including preview
- we can now use index.php to show a preview (stub)

2008-10-08

- many changes in page manager: we now have an almost working tree view that can expand and collapse

2008-10-01

- inserted a new permission: PERMISSION_ADMIN. If this bit is set in the user's permissions (either site, area or node), the user has at least one permission bit set to manipulate the site, area or node. It is more or less shorthand for 'one of the permission bits is set'.
- we now have an area menu in pagemanager

2008-09-30

- a few tweaks in adminlib.php (http-equiv headers)
- added a help topic in help button in navigation in adminlib.php

2008-09-29

- moved workhorse /program/admin.php to /program/lib/adminlib.php to avoid confusion with file names (which 'admin.php' do you mean?)
- further documented adminlib.php, tweaked the page layout

2008-09-26

- basic dispatcher in admin.php done
- change in datadefinition: added field job_permissions to user table
- change in datadefinition: added field high_visibility to user table
- added support for high visibility via additional stylesheet
- basic admin.php navigation is more or less done

2008-09-22

- further refined page layout for admin.php
- added a basic version checker at <http://siteatschool.org/version/index.php>

2008-09-21

- started with admin.php

2008-06-10

- added shortcuts for \$CFG->www and \$CFG->progwww

2008-05-28

- changed \$NODE, \$AREA and \$THEME into objects
- added support for redirecting users if they accidentally request a page with an external link
- made a start with a base class for themes
- started with a minimalistic theme called 'frugal'
- created start of a theme factory
- minor cosmetics

2008-05-09

- changed the creation of the \$DB global object: we now use a database factory

2008-04-29

- removed some of the trigger_error() calls in waslib/calculate_node(); only database errors now emit to screen if debug is TRUE; the rest now use logger(...,LOG_DEBUG) which writes to the database in itself.
- we now kick unauthorised users/passersby out of protected areas with a fatal error 070 (cannot find node in area). They should login the regular way IMHO. I re-used error code '070' on purpose in order not to leak information about protected areas.

2008-04-25

- changed name of config parameter for language from 'language' to 'language_key' to stay in sync with other (database) fields dealing with language codes.
- moved all code dealing with translations and languages to a separate file and into a special class.

2008-04-23

- cleaned up loginlib.php, added more documentation
- added logger() which logs important events in the database

2008-04-21

- done with t() and supporting routines
- added en and nl translations for loginlib.php

2008-04-16

- busy with translating strings via function t()

2008-04-11

- changed interface for last_insert_id() method: use table+field instead of sequencename
- added new table login_failures
- loginlib.php now also implements the blacklist feature and the failures counters

2008-04-09

- login mostly works, except blacklisting/failure counts
- added confirmation mail to user after succesful change of password

2008-04-08

- added a javascript alert in every login dialog (near the end)
- created a random password/laissez passer-generator

2008-04-07

- changed password requirements in loginlib: now min 6 chars etc.
- rearranged code in loginlib
- added more demodata to play with

2008-04-05

- first draft of WAS-logo incorporated in login dialog

2008-04-04

- still working on login/logout and various failure modes and also on the 2-step laissez-passer + bypass routine

2008-04-03

- still working on login/logout and various failure modes

2008-04-02

- started with login/logout routines, added file loginlib.php

2008-04-01

- rearranged some code
- we now always read the complete config table in core
- added more config parameters in demodata
- refined session handling code
- added if_exist_session() type of routine to make sure we are not fooled by spurious cookies

2008-03-30

- added tabledef for users, sessions
- added more support routines for database manipulation to waslib.php
- added a new file dbsessionlib.php

2008-03-19

- renamed db_quoted() to db_escape_and_quote() to make the effect more obvious
- we now know the \$node and the \$area and even the \$theme to use
- re-arranged some code; moved to waslib.php
- added even more demodata to 'play' with index.php and we added a clickable breadcrumb trail (that was _easy_).

2008-03-17

- renamed the quote() method to escape() in Database class because it may add confusion; the difference between escaping apostrofs and adding apostrofs around a string in order to insert into the database like in db_quote().
- added a simple test-routine to dump data from the database

2008-03-14

- added optional parameters \$limit and \$offset to Database->query() method
- we are now able to determine a valid area based on the user's request of 'area' and 'node'.

2008-03-13

- some more database manipulation routines added

2008-03-12

- evade the magic in magic_quotes_sybase by issuing a fatal error, circumvent the magic in magic_quotes_gpc() with function magic_unquote()
- changed the format of demodata from SQL-statements to nested arrays

2008-03-03

- added version check in init.php: the database version MUST match the code version, or else...

2008-02-29

- started with demodata.php to play with

2008-02-20

- added a simple tabledef dumper to install.php so we can see tabledefs in a tabluar form
- more tabledefinitions in tabledefs.php

2008-02-19

- added much to the main tabledefs.php

2008-02-16

- added a subtree under /program/install for tabledefs of the main system and also places for storing language specific demo data
- started with main tabledefs
- quick and dirty testcode in install.php to excercise/test the tabledefs

2008-02-15

- moved a lot of test code from install.php to /devel/test directory so I can 'play' with various subsystems without interfering with the real code. Currently /deve/test/test.php is focused on excercising the Database class.

2008-02-14

- added an extra dummy parameter to \$DB->last_insert_id() as hook for future extension with other database drivers
- added a function \$DB->table_exists()
- added a test with some 128 fielddefs in install.php (for now)

2008-02-12

- added \$DB->debug (default FALSE), can be set via constructor
- added \$DB->drop_table()
- added \$DB->create_table() using 'generic' (not MySQL-specific) datadefinition
- added another file to 'play' with: /program/install.php
- moved diff_microtime() to init.php

2008-02-11

- added new object type DatabaseResult to deal with queries returning data
- added a few more stubs in index.php to casually test Database/DatabaseResult-classes

2008-02-10

- documented \$CFG->debug

- got a few more methods working in mysql.class.php + q&d tests in index.php

2008-02-07

- experiments with the 'poor mans' database factory were more or less successful: the code works, but phpdoc fails badly when the same class is defined multiple times within the same package. However, a postgresql-driver in a separate @packages does work with phpdoc. Oh well.
- re-arranged some code in the Database-class: we now have a separate method to connect to the server and open the database, we no longer do that from the constructor. Also, we no longer bail out on MySQL-errors because we can expect errors during install and we don't want to bail out at that point.
- added error and errno to Database class which store the results from the underlying database, handy to find out what, exactly, went wrong.
- added the parameter \$CFG-debug in init.php
- moved the unset(\$CFG) from config(-example).php to index.php in order to make certain that we won't be fooled by a stray global set via \$_GET or whatever user input
- added a fake microtime() function in case it doesn't exist in the environment. The fake version uses a fractional part of 0.0 but is otherwise 100% compatible with the real microtime() string format. The reported time will be off by at most 2 seconds, oh well.
- Added a global object to record the performance, including the time of start and stop of the script execution. Could be used to identify bottlenecks.
- rearranged the order of code in init.php
- added error_reporting depending on the value of \$CFG->debug.
- brought the call to the new Database class in line with the new class definition
- removed the calls to error_exit() from the Database class; they do not belong there.
- documented the format of the release date variable in version.php to include the full seconds-resolution ISO 8601 date
- added more error checking in Database class via \$DB->errno and \$DB->error

2008-02-06

- made a start with databaselib.php and mysql.class.php
- some tweaking in version.php: now record full release date+time

2008-02-04

- re-arranged the comments in config-example.php, hopefully clarifies

2008-02-01

- added init.php
- added version.php: keeps track of version of php-files (for comparison with database to determine necessary update actions)

2008-01-31

- added preliminary versions of README, INSTALL, etc.
- completed a quick and dirty script to generate documentation with phpdoc
- completed a script to generate both distribution version and development version of W@S

2008-01-28

- committed a first version of /index.php and config-example.php to

CVS; we have now actually started creating program code

CREDITS

/program/CREDITS.txt

\$Id: CREDITS.txt,v 1.14 2016/06/28 13:21:16 peter Exp \$

For an up to date overview of credits, please visit the
project website at <<http://websiteatschool.eu>>.

Core team

=====

- * Karin Abma (ICT coordinator of the Public Primary School Rosa Boekdrukker in Amsterdam, The Netherlands).
- * Peter Fokker (Ingenieursbureau PSD, main developer, programmer).
- * Dirk Schouten (R.I.P.) (former teacher, user manuals writer).

Translators

=====

- * Said Taki (Arabic, under construction).
- * Boyan Kirchev (Bulgarian).
- * Jing Fang Liu (Chinese).
- * Danny Yen (Chinese).
- * Christian Borum Loebner - Olesen (Danish).
- * Core team (Dutch).
- * Core team (English).
- * A. Darvishi (Farsi, under construction).
- * Abigail Hieselaar (Filipino).
- * Laura RÃ¥man (Finnish).
- * Jean Peyratout (French).
- * Marjolaine Audoux (French).
- * David (German).
- * S. Stadoll (German).
- * Fabienne Kudzielka (German).
- * Claudia GÃ¶hnert (German).
- * Iakovos Christoforidis (Greek).
- * Erika Swiderski (Hungarian).
- * Gergely Sipos (Hungarian).
- * Ernst FrankemÃ¶lle (Italian).
- * Giovanni Thomas (Papiamento).
- * Waldemar Pankiw (Polish).
- * Rita Valente Ribeiro da Silva (Portugese).
- * Thais Rizzi (Portugese).
- * Loana Lenghel (Romanian).
- * Anastassia Blechko (Russian).
- * Sladjana Cetin (Serbo-Croatian).
- * Anouk Coumans (Spanish).
- * Margot Molier (Spanish).
- * Hannah Tulleken (Spanish).
- * Teresa Villanova (Spanish).

- * Hansje Cozijnsen (Swedish).
- * ÃœlkÃ¼ Gaga (Turkish, under construction).
- * Nasira Parveen (Urdu).
- * Quynh Nguyen (Vietnamese).
- * Piet Damsma (Western Frisian).
- * Galina Valina (Bulgarian).
- * P.M. Tabrizi (Farsi, under construction).
- * Rosanna Gaddoni (Italian).
- * Erdil Oral (Turkish, under construction).
- * Thao Doan (Vietnamese).
- * Rashmi Bookanakere (Hindi, under construction).
- * Nikoleta Koncikova (Slovak, under construction).

Translators (Website@School Manual)

=====

- * Boyan Kirchev (Bulgarian, under construction).
- * Denitza Lambreva (Bulgarian, under construction).
- * Karin Abma (Dutch, under construction).
- * Rieks van Rooijen (Dutch, under construction).
- * Tamara Zwaan (Dutch, under construction).
- * Jean Peyratout (French, under construction).
- * Marjolaine Audoux (French, under construction).
- * Giovanni Thomas (Papiamentu, under construction).
- * Keli Tracz (Portugese, under construction).
- * Natalia Kempel (Russian, under construction).
- * Anouk Coumans (Spanish, under construction).
- * Gala LazÃ¡ro Mur (Spanish, under construction).
- * Hansje Cozijnsen (Swedish, under construction).

Code contributions

=====

- * Frederico Caldeira Knabben (CKEditor and FCKeditor, see <http://ckeditor.com>).
- * Ger Versluis (HV Menu, used in the Rosalina theme).

Graphics

=====

- * Micky Faas (Website@School logos).
- * Greg Whitaker (Some icons).
- * Hans Vissers (photo of Theobroma cacao)

Donations

=====

- * Europees Platform voor het Onderwijs (European Platform for Education).
- * Stichting KBA Nieuw West (Foundation Catholic Primary Education, Amsterdam).
- * Openbare Basisschool Rosa Boekdrukker (Public Primary School Rosa Boekdrukker, Amsterdam).
- * Nederlandse Vereniging voor Blinden en Slechtienden (Dutch Association for Blind and Visually Impaired).
- * Stichting Blinden-Penning Foundation for activities for blind and visually impaired.

- * Stichting Mijn CO2Spoor
(MyCO2Track Foundation).
- * Enablement.
- * Lemstra Techniek.
- * Stichting Internet Domeinregistratie Nederland (SIDN)
(Foundation for Internet Domain Registration in the Netherlands)
- * Harm Hofstede.
- * C. van OrlÃ©.
- * Steunpunt ICT.
- * Volkshogeschool Eerbeek.
- * EURICT.
- * OMBS ZieZo.
- * John F. Kennedyschool, Breda.
- * M. Heeman.
- * Stichting EDICT.
- * RKB De Hoeksteen, Bussum.
- * Stichting Blindenhulp, 's-Gravenhage.
- * Landelijke Stichting voor Blinden en Slechtzienden, Utrecht.
- * Rotterdamse Stichting Blindenbelangen, Rotterdam.
- * Stichting tot Verbetering van het Lot der Blinden.
- * Het Gewilde Westen.
- * RoadVision Traffic.
- * Many anonymous donors.

Testers

=====

- * J.G.M. Meijer.
- * Hans Wolters.
- * Stefan Schurtz.
- * Jan Hertog (R.I.P.).
- * Rieks van Rooijen.

Others

=====

- * Carla Alma.
- * Margret Kwantes.
- * Matthijs Koopmans.
- * Rod Fernandes.
- * Ernst FrankemÃ¶lle.

Please contact us if you feel your name should be mentioned here.

Appendix D - Todo List

In Package wascore

In [acceptable_new_password\(\)](#)

- Entropy! Perhaps it is better to value the strength of a password in a different way, say 'CorrectHorseBatteryStaple' could be better than 'Tr0ub4dor&3'. Also, why force a limited set of ASCII-characters [0-9A-Za-z] when there is so much entropy to gain by using characters in the range U+0080 - U+10FFFF? (20140912/PF)

In [AcIManager](#)

- there is something not right with buffering the tabledefs. If an error occurs, we get FALSE instead of an array. Mmmmm....

In [AdminOutput](#)

- add a 'funnel mode': disable all distracting links that could seduce the user to leave and leave locked records (eg. nodes)
- carefully check if we need more headers in html-head section of document, see AdminOutput().

In [AdminOutput::AdminOutput\(\)](#)

- do we need a link rel="shortcut icon" type of header too?
- do we really need more meta-headers?
- is it really wise to add a base header? It interferes with the session cookie whenever you login at another URL than the base+'admin.php'... Comment it out for now

In [AreaManager](#)

- we need to take care of spurious spaces in inputs (or do we?)

In [AreaManager::area_delete\(\)](#)

- since multiple tables are involved, shouldn't we use transaction/rollback/commit? Q: How well is MySQL suited for transactions? A: Mmmmm.... Which version? Which storage engine?

In [AreaManager::area_overview\(\)](#)

- should we add a paging function to the list of areas? Currently all areas are shown in a single list...
- should we make two categories: 'public' and 'private' in the list of areas? Maybe handy when there are many manu areas, but it would be inconsistent with the page manager menu which simply lists the areas in the sort order. Easy way out: the user is perfectly capable to set the sort order in such a way that the sort order already groups the public and private areas. Oh well....

In [AreaManager::area_setdefault\(\)](#)

- should we acknowledge the changed default to the user or is it enough to see the icon 'move'?
- should we send alerts? If so, can we use the routine to queue messages from pagemanager? A reason not to send alerts: the alerts will be sent as soon as a page is added to the new area, so why bother?

In [PageManager::calculate_updated_sort_order\(\)](#)

- Clean up this code, it is very hairy

In [Theme::calc_breadcrumb_trail\(\)](#)

- split into two separate routines, one to set the tree, another to construct the list of anchors

In [DatabaseMysql::column_definition\(\)](#)

- 'enum' type equivalent with varchar, enum_values[] array is not used at all, only as a form of documentation
- should we allow both int and integer?

In [DatabaseMysqli::column_definition\(\)](#)

- 'enum' type equivalent with varchar, enum_values[] array is not used at all, only as a form of documentation
- should we allow both int and integer?

In [DatabaseMysql::concat\(\)](#)

- perhaps extend this function to accept more than 2 strings?

In [DatabaseMysql::concat\(\)](#)

- perhaps extend this function to accept more than 2 strings?

In [ConfigAssistant](#)

- implement checklist

In [DatabaseMysql::connect\(\)](#)

- weigh pros and cons of persistent database connections, perhaps add as config option?

In [DatabaseMysql::connect\(\)](#)

- weigh pros and cons of persistent database connections, perhaps add as config option?

In [convert_to_type\(\)](#)

- perhaps change the possible values of \$type to full strings rather than 'cryptic' single letter codes. Furthermore: what do we do with invalid dates, times and date/times? For now it is a stub, returning \$value as-is. Oh well.

In [DatabaseMysql::create_table\(\)](#)

- document correct link for documentation of generic table definition 'tabledefs.php'

In [DatabaseMysqli::create_table\(\)](#)

- document correct link for documentation of generic table definition 'tabledefs.php'

In [DatabaseMysql::create_table_sql\(\)](#)

- document correct link for documentation of generic table definition 'tabledefs.php'
- find a way to deal with the enum values: where do we keep them? Or do we keep them at all?

In [DatabaseMysqli::create_table_sql\(\)](#)

- document correct link for documentation of generic table definition 'tabledefs.php'
- find a way to deal with the enum values: where do we keep them? Or do we keep them at all?

In [database_factory\(\)](#)

- perhaps add postgresql in a future version

In [dbsession_close\(\)](#)

- should we do something with locking the session record from `dbsession_open()` until `dbsession_close()`? For now, the session record is not locked in any way, so the latest call gets to keep its changes Mmmm....

In [`dbsession_create\(\)`](#)

- should we also record the IP-address of the user in the session record? In a way this is a case of information leak, even though it is only between authenticated users. Mmmm...

In [`dbsession_remove_obsolete_sessions\(\)`](#)

- this routine should not need to know about nodes and stuff. Mmmm... remove FK constraint?

In [`GroupManager::delete_group_capacities_records\(\)`](#)

- since multiple tables are involved, shouldn't we use transaction/rollback/commit? Q: How well is MySQL suited for transactions? A: Mmmmm.... Which version? Which storage engine?

In [`dialog_get_label\(\)`](#)

- if we let the hotkey from the label prevail and add it to the input tag, why add a hotkey to the label too?

In [`dialog_get_widget\(\)`](#)

- we could manipulate the title attribute of input strings like "please enter a number between {MIN} and {MAX}" based on the various value properties instead of just displaying the title. oh well, for a future version, perhaps...
- we now only cater for buttons via input type="submit" without the option to visualise the accesskey. Using the button tag could solve that, but button is not defined before HTML 4.01. What to do?

In [dialog_get_widget_file\(\)](#)

- if we let the hotkey from the label prevail and add it to the input tag, why add a hotkey to the label too?
- should we do something with an um-empty \$value? If so, waht? The browser ignores this...

In [dialog_get_widget_richtextinput\(\)](#)

- if we let the hotkey from the label prevail and add it to the input tag, why add a hotkey to the label too?

In [dialog_get_widget_textinput\(\)](#)

- if we let the hotkey from the label prevail and add it to the input tag, why add a hotkey to the label too?

In [AclManager::dialog_tableform\(\)](#)

- bailing out on non-array is a crude way of error handling: this needs to be fixed

In [dialog_validate\(\)](#)

- add an error message to

In [error_exit\(\)](#)

- Q: do we really want to 'leak' a link to the main site?

In [FileManager::FileManager\(\)](#)

- a nice filter for JOB_IMAGEBROWSER and also an alternative user interface for browsing/selecting

images

In [ConfigAssistant::get_dialogdef\(\)](#)

- implement checklist

In [PageManager::get_dialogdef_add_node\(\)](#)

- should we make the defaults in this routine configurable? (I'm not convinced they should)

In [AclManager::get_dialogdef_admin\(\)](#)

- handle the related information in this dialog

In [AclManager::get_dialogdef_intranet\(\)](#)

- handle the related information in this dialog

In [TranslateTool::get_dialogdef_language_domain\(\)](#)

- try to figure this out: when the delimiter in \$name was a dot '.' \$_POST contained a '_' instead. WTF? (it seems that a colon works... for now)

In [get_editor_names\(\)](#)

- retrieve this list from 'config'-table?

In [AreaManager::get_icon_delete\(\)](#)

- should we check to see if the area is empty before showing delete icon? Or is it soon enough to refuse deletion when the user already clicked the icon? I'd say the latter. For now...

In [PageManager::get_icon_delete\(\)](#)

- should we display trash can icons for sections with non-empty subsections? there really is no point, because we eventually will not accept deletion of sections with grandchildren in task_node_delete. Hmmmmm..... For now I just added the condition that access is denied when a section has grandchildren. Need to refine this, later. Also, how about readonly nodes? Surely those cannot be deleted... should it not show in the icon?

In [PageManager::get_icon_page_preview\(\)](#)

- if this is a public area, the user can see every page, except the expired/embargo'ed ones should we take that into account too? I'd say that is way over the top. How about pages in an intranet where the user has view privilege? Complicated. KISS: only show preview to those that can edit or edit content.

In [Theme::get_logo\(\)](#)

- should we take path_info into account here too???? how about /area/aaa/node/nnn instead of /aaa/nnn???

In [Language::get_phrase\(\)](#)

- should we return an error for an invalid specific language?

In [GroupManager](#)

- Perhaps this class should be merged with the UserManager class because there is a lot of overlap. Mmmmm.... maybe in a future refactoring operation.

In [GroupManager::group_delete\(\)](#)

- since multiple tables are involved, shouldn't we use transaction/rollback/commit? Q: How well is MySQL suited for transactions? A: Mmmmm.... Which version? Which storage engine?

In [GroupManager::group_savenew\(\)](#)

- maybe we should find a more elegant way to check a field for uniqueness
- should we delete the datadirectory if something goes wrong?

In [Useraccount::has_module_node_permissions\(\)](#)

- FixMe: we need to take the parent nodes into account too!

In [Useraccount::has_node_permissions\(\)](#)

- FixMe: we need to take the parent nodes into account too!

In [href\(\)](#)

- should we merge this with `html_a()` and/or rename this routine to `html_href()`?

In [install_module\(\)](#)

- we should refactor and combine `install_theme()` and `install_module()`

In [install_theme\(\)](#)

- we should refactor and combine `install_theme()` and `install_module()`

In [is_expired\(\)](#)

- this function also 'repairs' circular references. This should move to a separate function but for the time being it is "convenient" to have automatic repairs... tree-repair

In [is_under_embargo\(\)](#)

- this function also 'repairs' circular references. This should move to a separate tree-repair function but for the time being it is "convenient" to have automatic repairs...

In [job_start\(\)](#)

- this routine is a stub

In [job_statistics\(\)](#)

- this routine is a little bit too long, it should be split into smaller chunks

In [job_tools\(\)](#)

- fix permissions for backup tool! perhaps another bit?

In [TranslateTool::languages_overview\(\)](#)

- should we add a paging function to the (perhaps loooooong) list of languages?

In [lock_record\(\)](#)

- do we need a 'force lock' option to forcefully take over spurious locks? Maybe guru can override?
- perhaps we can save 1 trip to the database by checking for something like `UPDATE SET locked_by = $session_id WHERE (id = $id) AND ((locked_by IS NULL) OR (locked_by = $session_id))` but I don't know how many affected rows that would yield if we already had the lock and effectively nothing changes in the record. (Perhaps always update atime to force 1 affected row?)

In [logger\(\)](#)

- should we make this configurable and maybe log directly to syslog (with automatic logrotate) or do we want to keep this 'self-contained' (the webmaster can read the table, but not the machine's syslog)?

In [loginlib.php](#)

- should we normalize the remote_addr everywhere? We now rely on the remote_addr being equal to some stored value (in the database) but with an IPv6 address there are several possibilities to have different representations of the same address (e.g. '::dead:beef' is equivalent to '::0:dead:beef' or even '::DeAd:BeeF' or '0000:0000:0000:0000:0000:0000:DEAD:BEEF'. This problem also exists with IPv4: '127.0.0.1' is equivalent to '127.000.000.001'. *sigh*
- should we suppress the username in the laissez-passer routine? We _do_ leak the the username in an insecure email message. This does require making the laissez-passer code unique in the database (currently only username+code has to be unique and that's easy because the username itself is unique).

In [login is blacklisted\(\)](#)

- Should we first change \$remote_addr to canonical form for good comparison? (20140912/PF)

In [login propagate navigation\(\)](#)

- Maybe we could insert the parameters from \$add into \$path_info. OTOH: adding the existing way always works. If it aint broken... (2014-10-09/PF)

In [main admin\(\)](#)

- should we cater for a special 'print' button + support for a special style sheet for media="print"?

In [main cron.php](#)

- should we act differently based on how we are called (cli or web), eg send more queued alerts?
- should we build extra access protection via passwords and/or IP-addresses?

In [main_index\(\)](#)

- cleanup login/logout-code

In [main_index.php](#)

- add the performance results in a HTML-comment if not CFG->debug, in sight otherwise

In [PageManager::module_connect\(\)](#)

- should we pass the area_id at all? What happens when a node is moved to another area without informing the module? Questions, questions, questions...

In [PageManager::module_disconnect\(\)](#)

- should we pass the area_id at all? What happens when a node is moved to another area without informing the module? Questions, questions, questions...

In [PageManager::module_load_admin\(\)](#)

- should we sanitise the modulename here? It is not user input, but it comes from the modules table in the database. However, if a module name would contain sequences of "../" we might be in trouble

In [module_load_view\(\)](#)

- should we sanitise the modulename here? It is not user input, but it comes from the modules table in the database. However, if a module name would contain sequences of "../" we might be in trouble

In [password_hash_check\(\)](#)

- Now if the sha1-hash was compressed into 5 or 6 bits/char instead of hexadecimal digits this routine would fail miserably, so don't do that.

In [performance_get_seconds\(\)](#)

- maybe we should get rid of this \$PERFORMANCE object, because it doesn't do that much anyway

In [quoted_printable\(\)](#)

- should we change the code to accomodate the canonical newline CRLF in the input?

In [Email::rfc2047_qstring\(\)](#)

- maybe optimise this routine to let pure ASCII-words through unencoded (in a later version)

In [Email::rfc5322_message_id\(\)](#)

- how about UTF-8 hostnames? Mmmm...

In [sanitise_filename\(\)](#)

- should we check for overlong UTF-8 encodings: C0 AF C0 AE C0 AE C0 AF equates to ../ or is that dealt with already by filtering on letters/digits and embedded dots/dashes/underscores?

In [AclManager::save_data_admin\(\)](#)

- fix the crude error check on dialogdef === FALSE here

In [PageManager::save_node\(\)](#)

- there is something wrong with embargo: should we check starting at parent or at node? this is not clear: it depends on basic/advanced and whether the embargo field changed. mmmm... safe choice: start at node_id for the time being
- this routine could be improved by refactoring it; it is too long!

In [Email::set_header\(\)](#)

- should we bring the Capi-Tali-Sation of \$name in line with the default capitalisation in the list above?

In [FileManager::show_dialog_add_files\(\)](#)

- should we take the locale into account formatting numbers (thousands separator)?

In [FileManager::show_directories_and_files\(\)](#)

- This routine is way too long, it should be split up into smaller subroutines

In [FileManager::sort_entries\(\)](#)

- it is a pity I cannot reference \$this->sort from within the 6 cmp-functions...

In [task_logview\(\)](#)

- should we allow for fancy selection mechanisms on the logfile or is that over the top?

In [PageManager::task_node_delete\(\)](#)

- should we display trash can icons for sections with non-empty subsections in treeview? there

really is no point, because we eventually will not accept deletion of sections with grandchildren. Hmmmmm.....

In [PageManager::task_page_preview\(\)](#)

- the check on permissions can be improved (is PERMISSION_XXXX_EDIT_NODE enough?)
- there is an issue with redirecting to another site: officially the url should be fully qualified (ie. \$CFG->www). I use the shorthand, possibly without scheme and hostname (\$CFG->www_short). This might pose a problem with picky browsers. See [calculate uri shortcuts](#) for more information.

In [PageManager::task_save_newnode\(\)](#)

- about 'sort_order': do we insert nodes at the end or the beginning of a parent section?
- how do we alert users that an embargo date has come around? Do we schedule alerts via cron?

In [theme_factory\(\)](#)

- should we massage the directory and file names of the included theme?
- what if the theme is not found? Currently no alternative is loaded but FALSE is returned.

In [token_create\(\)](#)

- Should we enforce valid UTF8 in \$reference and \$ip_addr? We might have substr() trouble...

In [token_garbage_collect\(\)](#)

- This routine should be called from cron.php every once in a while

In [tree_build\(\)](#)

- repairing a node doesn't really belong here, in this routine. we really should have a separate 'database repair tool' for this purpose. someday we'll fix this....
- what if we need the trees of two different areas? should the static var here be an array, keyed by area_id?

In [tree_visibility\(\)](#)

- how about making all nodes under embargo visible when previewing a page or at least the path from the node to display?

In [FileManager::unique_filename\(\)](#)

- Should we take care of the race condition in this routine? Should we already create an empty file or is that clutter?

In [update_statistics\(\)](#)

- maybe extend this routine to actually store more statistics information in a separate table

In [UserManager](#)

- Perhaps this class should be merged with the GroupManager class because there is a lot of overlap. Mmmmm.... maybe in a future refactoring operation.

In [UserManager::user_delete\(\)](#)

- since multiple tables are involved, shouldn't we use transaction/rollback/commit? Q: How well is MySQL suited for transactions? A: Mmmmm.... Which version? Which storage engine?

In [UserManager::user_savenew\(\)](#)

- maybe we should find a more elegant way to check a field for uniqueness

- shouldn't we end with the edit-user dialog rather than the users overview? that might make more sense...

In [FileManager::valid_path\(\)](#)

- the check on '/../' is inconclusive if the \$path is encoded in UTF-8: the overlong sequence 2F C0 AE 2E 2F eventually yields 2F 2E 2E 2F or '/../'. Reference: RFC3629 section 10.

In [FileManager::virusscan\(\)](#)

- maybe use MIME for sending alert if not 7bit message?
- This routine is quite *nix-centric. I'm not sure how this would work other server platforms. Should we do something about that?

In [Zip::zip_add_data\(\)](#)

- should we handle the option of a better compression level (eg. level 9) in gzcompress()? we could check to see if CMF equals 0x78 and FLG is either 0x01, 0x5E, 0x9C or 0xDA the latter 4 values might have an effect on general purpose bit flag bits 2 and 3. for now we'll just keep it simple, but there might be a little something to improve here.
- should we handle the possibility of an additional 4 bytes for DICTID (RFC1950, reference [2])?

In Package wasmod_aggregator

In [aggregator_check_node_list\(\)](#)

- perhaps we should check more thoroughly for node existence but that implies also checking for user access etc. etc. We postpone that to later when the visitor's credentials will be checked against the list of nodes. So: a syntax check only. Sort of.

In [aggregator_view\(\)](#)

- what to do with htmlpages containing h1's and h2's? Get rid of those? Mmmm....

In Package wastheme_axis

In [axis_upgrade\(\)](#)

- maybe make this a little less quick and dirty?

In Package wasmod_confab

In [confab_braille_show_all\(\)](#)

- this looks a lot like the [confab_braille_show_main\(\)](#), maybe there is room for improvement.

In [confab_join_dialog_validate\(\)](#)

- We may have to enforce usernames cannot end in an asterisk in Account Manager.

In [confab_show_edit_moderation\(\)](#)

- this routine is too long

In Package wasmod_crew

In [crew_search\(\)](#)

- should we take status=0 into account too (individual entry to the workshop)? it looks like a lot of effort for a low yield

In [crew_view\(\)](#)

- FixMe: we need to take parent node permissions into account as soon as we can assign crew permissions to sections. We now specifically look at permissions in table acls_modules_nodes OR at global (guru) permissions for modules in table acls. (June 2013).
- FixMe: we should rename SaveEdit and Save to Save and Done (May 2014)
- FixMe: we should somehow add a csrftoken into the mix (May 2014)

In Package wasinstall

In [InstallWizard::check_compatibility\(\)](#)

- add more tests, e.g. for gd, safe_mode, memory limit, etc.

In [demodata_users_groups\(\)](#)

- get rid of the \$wizard kludge!
- should we also add groups_capacities, acls, users_groups_capacities to \$config or are users and groups enough?
- should we append an underscore to the userpaths to make sure we don't clash with the first user account?

In [InstallWizard::get_default_install_values\(\)](#)

- should we check the program version versus the stored program version here?
- there is something wrong with the default for \$cms_www; FIXME (commented out for now)

In [InstallWizard::get_page\(\)](#)

- should we promote language and high_visibility to function parameters instead of using \$_SESSION directly?

In [install.php](#)

- how prevent third party-access to install.php after initial install? .htaccess? !exists(../config.php)?
- we should make sure that autosession is disabled in php.ini, otherwise was won't work
- we should make sure that register globals is off
- we should make sure that we can actually set cookies (necessary when logging in).

In [InstallWizard::perform_installation\(\)](#)

- should we save the config.php to the datadir if the main dir fails? Mmmm.... security implications?
- this routine badly needs refactoring

In [InstallWizard::save_cms\(\)](#)

- also take safe_mode into account? Should that be a requirement for succesfull installation?

In [InstallWizard::show_dialog_cms\(\)](#)

- can we suppress even more fields here in case of a Standard installation?

In [InstallWizard::show_dialog_compatibility\(\)](#)

- more tests to perform here: safe mode, memory limit, processing time limit, register globals

In [tabledefs.php](#)

- automatically create appropriate sequence name for serial fields??? or add seqdefs too?

In [InstallWizard::write_config_php\(\)](#)

- should we make the filemode (hardcoded at 0400) configurable/customisable?

In Package wasmod_htmlpage

In [htmlpage_cron.php](#)

- change this stub into a real cron function.

In [htmlpage_get_dialogdef\(\)](#)

- should we add a title here?

In [htmlpage_show_edit\(\)](#)

- get rid of 'version' field and related cruft (or do something useful with it). 2014-05-05/PF

In Package wasmod_mailpage

In [mailpage_send_message\(\)](#)

- extra validation of set_mailreplyto and set_subject?
- make body of mail configuratble?
- more available parameters in subject_line?

In [mailpage_show_thankyou\(\)](#)

- should we have an OK button at all???

In Package wasmod_mypage

In [mypage_profile_get_dialogdef\(\)](#)

- should we remove username and datadir altogether?

In Package wasmod_newsletter

In [newsletter_demodata\(\)](#)

- There are various stubs that need to be fixed.

In [newsletter_subscriptions_save\(\)](#)

- what to do with equal email addresses: can those simply be added or must they be unique?

In [newsletter_urls2cids\(\)](#)

- we may be doing double work by searching/replacing the same url()'s again and again

In [newsletter_view_contribution_send\(\)](#)

- extra validation of set_mailreplyto and set_subject?
- make body of mail configuratble?
- more available parameters in subject_line?

In [newsletter_view_contribution_thankyou\(\)](#)

- should we have an OK button at all???

In Package wastheme_rosalina

In [ThemeRosalina::get_logo\(\)](#)

- should we take path_info into account here too???? how about /area/aaa/node/nnn instead of /aaa/nnn???

In [rosalina_upgrade\(\)](#)

- maybe make this a little less quick and dirty?

In Package wastheme_schoolyard

In [schoolyard_upgrade\(\)](#)

- maybe make this a little less quick and dirty?

In Package wasmod_search

In [search_get_qwords\(\)](#)

- better parse of keywords taking quotes into account too.

In [search_highlight\(\)](#)

- is there a chance that our static cache \$q fails if somehow a different \$qwords is used on subsequent calls? Mmmmm...

- what to do with overlapping qwords, eg. "foo" and "foobar"? result depends on search/replace execution order: either "**foobar**" or "**foobar**"

In [search_view\(\)](#)

- should we use the token routines here too?

In Package wasmod_snapshots

In [SnapshotViewer::add_snapshot_navbar\(\)](#)

- clean up this ugly code

In [SnapshotViewer::get_snapshots_configuration\(\)](#)

- check for information leaks (private path) here?

In [SnapshotViewerInline::run\(\)](#)

- check permissions (ACL) to prevent leaking a private area path to anonymous visitors?

In [SnapshotViewer::run\(\)](#)

- check permissions (ACL) to prevent leaking a private area path to anonymous visitors?

In [snapshots_check_path\(\)](#)

- should the user / group really be active here? If not, the images will fail in file.php but that may leak information about inactive users. Hmm...
- we should use a different error message as soon as it is available in was.php, eg.

'validate_bad_directory' (much like 'validate_bad_filename').

In [snapshots show edit\(\)](#)

- we might want to jazz up this dialog by adding some sort of 'directory browser' for the snapshots_path field using a pop-up window. Mmmmm.... future refinements...

Index

A

aggregator.php	726
/program/modules/aggregator/languages/pap/aggregator.php	
axis.php	733
/program/themes/axis/languages/pap/axis.php	
admin.php	722
/program/languages/pap/admin.php	
axis.php	711
/program/themes/axis/languages/it/axis.php	
aggregator.php	704
/program/modules/aggregator/languages/it/aggregator.php	
admin.php	742
/program/languages/pl/admin.php	
axis.php	748
/program/themes/axis/languages/pl/axis.php	
admin.php	773
/program/languages/ro/admin.php	
axis.php	766
/program/themes/axis/languages/pt/axis.php	
aggregator.php	759
/program/modules/aggregator/languages/pt/aggregator.php	
admin.php	755
/program/languages/pt/admin.php	
admin.php	700
/program/languages/it/admin.php	
axis.php	693
/program/themes/axis/languages/hu/axis.php	
axis.php	648
/program/themes/axis/languages/fr/axis.php	
aggregator.php	641
/program/modules/aggregator/languages/fr/aggregator.php	
admin.php	637
/program/languages/fr/admin.php	
admin.php	630
/program/languages/fil/admin.php	
admin.php	657
/program/languages/fry/admin.php	
aggregator.php	661
/program/modules/aggregator/languages/fry/aggregator.php	
admin.php	687
/program/languages/hu/admin.php	
admin.php	678
/program/languages/hi/admin.php	
axis.php	670
/program/themes/axis/languages/fry/axis.php	

althing.php	662
/program/modules/althing/languages/fry/althing.php	
axis.php	779
/program/themes/axis/languages/ro/axis.php	
admin.php	785
/program/languages/ru/admin.php	
aggregator.php	894
/program/modules/aggregator/languages/vi/aggregator.php	
althing.php	895
/program/modules/althing/languages/vi/althing.php	
admin.php	890
/program/languages/vi/admin.php	
axis.php	881
/program/themes/axis/languages/ur/axis.php	
aggregator.php	874
/program/modules/aggregator/languages/ur/aggregator.php	
axis.php	903
/program/themes/axis/languages/vi/axis.php	
admin.php	912
/program/languages/zh/admin.php	
aggregator_connect()	924
connect this module to a node	
aggregator_check_node_list()	923
validate and massage the user-supplied node list	
aggregator_admin.php	923
/program/modules/aggregator/aggregator_admin.php - management interface for aggregator-module	
axis.php	918
/program/themes/axis/languages/zh/axis.php	
admin.php	870
/program/languages/ur/admin.php	
axis.php	865
/program/themes/axis/languages/tr/axis.php	
admin.php	808
/program/languages/sk/admin.php	
admin.php	803
/program/languages/sh/admin.php	
axis.php	796
/program/themes/axis/languages/ru/axis.php	
aggregator.php	789
/program/modules/aggregator/languages/ru/aggregator.php	
aggregator.php	812
/program/modules/aggregator/languages/sk/aggregator.php	
agplv3_compliance()	820
admin.php	861
/program/languages/tr/admin.php	
axis.php	853
/program/themes/axis/languages/sv/axis.php	
aggregator.php	846
/program/modules/aggregator/languages/sv/aggregator.php	
admin.php	842
/program/languages/sv/admin.php	
axis.php	625

/program/themes/axis/languages/fi/axis.php	619
admin.php	619
/program/languages/fi/admin.php	
AreaManager::get_dialogdef_edit_area()	254
<i>construct the edit area basic properties dialog</i>	
AreaManager::get_dialog_data()	254
<i>fill the dialog with current area data from the database</i>	
AreaManager::get_dialogdef_delete_area()	254
<i>construct the delete area dialog</i>	
AreaManager::get_dialogdef_add_area()	253
<i>construct the add area dialog</i>	
AreaManager::count_existing_theme_properties()	253
<i>determine the number of existing properties for a theme in an area</i>	
AreaManager::get_icon_delete()	254
<i>construct a clickable icon to delete this area</i>	
AreaManager::get_icon_edit()	255
<i>construct a clickable icon to edit theme properties of this area (edit advanced)</i>	
AreaManager::reset_theme_defaults()	256
<i>reset the theme properties of an area to the default values</i>	
AreaManager::get_theme_records()	256
<i>retrieve a list of all available theme records</i>	
AreaManager::get_options_themes()	256
<i>fetch a list of themes available for an area</i>	
AreaManager::get_icon_home()	255
<i>construct a clickable icon to set the default area</i>	
AreaManager::a_param()	253
<i>shorthand for the anchor parameters that lead to the area manager</i>	
AreaManager::area_setdefault()	252
<i>make the selected area the default for the site</i>	
AreaManager::area_delete()	249
<i>delete an area from this site after confirmation</i>	
AreaManager::area_add()	248
<i>present a dialog where the user can enter minimal properties for a new area</i>	
AreaManager::\$show_parent_menu	248
AreaManager::\$output	248
AreaManager::area_edit()	249
<i>show the basic properties edit dialog and the edit menu</i>	
AreaManager::area_edittheme()	250
<i>show the theme/area configuration dialog and the edit menu</i>	
AreaManager::area_savenew()	252
<i>save the newly added area to the database</i>	
AreaManager::area_save()	252
<i>validate and save modified data to database</i>	
AreaManager::area_resettheme()	251
<i>reset the theme configuration to the factory defaults</i>	
AreaManager::area_overview()	250
<i>display list of areas with edit/delete icons etc. and option to add an area</i>	
AreaManager::show_edit_menu()	257
<i>display the edit menu via \$this->output</i>	
AreaManager::show_parent_menu()	257
<i>allow the caller to use the menu area (or not)</i>	
althing.php	574
/program/modules/althing/languages/el/althing.php	

aggregator.php	573
/program/modules/aggregator/languages/el/aggregator.php	
admin.php	569
/program/languages/el/admin.php	
axis.php	560
/program/themes/axis/languages/de/axis.php	
axis.php	580
/program/themes/axis/languages/el/axis.php	
admin.php	589
/program/languages/es/admin.php	
axis.php	612
/program/themes/axis/languages/fa/axis.php	
admin.php	606
/program/languages/fa/admin.php	
axis.php	597
/program/themes/axis/languages/es/axis.php	
aggregator.php	593
/program/modules/aggregator/languages/es/aggregator.php	
aggregator.php	553
/program/modules/aggregator/languages/de/aggregator.php	
admin.php	549
/program/languages/de/admin.php	
axis.php	508
/program/themes/axis/languages/ar/axis.php	
ar_manifest.php	503
/program/languages/ar/ar_manifest.php - description of the Arabic translation	
admin.php	502
/program/languages/ar/admin.php	
AreaManager::sort_order_new_area()	257
determine the value for the sort order of a new area	
admin.php	515
/program/languages/bg/admin.php	
aggregator.php	519
/program/modules/aggregator/languages/bg/aggregator.php	
axis.php	542
/program/themes/axis/languages/da/axis.php	
admin.php	536
/program/languages/da/admin.php	
axis.php	527
/program/themes/axis/languages/bg/axis.php	
althing.php	520
/program/modules/althing/languages/bg/althing.php	
aggregator_disconnect()	924
disconnect this module from a node	
aggregator_get_dialogdef()	925
construct a dialog definition for the aggregator configuration data	
althing_uninstall()	961
uninstall the module	
althing_upgrade()	962
upgrade the module	
althing_install()	961
install the module	
althing_demodata()	960

<i>add demonstration data to the system</i>	
althing_install.php	960
<i>/program/modules/althing/althing_install.php - installer of the althing module</i>	
althing_manifest.php	963
<i>/program/modules/althing/althing_manifest.php - description of the althing module</i>	
althing_search.php	964
<i>/program/modules/althing/althing_search.php - interface to the search-part of this module</i>	
ALTHING_DIRNAME	966
althing_view.php	966
<i>/program/modules/althing/althing_view.php - interface to the view-part of the althing module</i>	
althing_search_results()	965
<i>count the number of hits in \$table</i>	
althing_search()	964
<i>search this module's content in selected nodes for keywords in \$qwords</i>	
althing_cron()	959
<i>routine that is called periodically by cron</i>	
althing_cron.php	959
<i>/program/modules/althing/althing_cron.php - interface to the cron-part of the althing module</i>	
ALTHING_HARVEST_LIMIT	954
ALTHING_DEFAULT_VISIBILITY_VISIBLE	954
ALTHING_DEFAULT_VISIBILITY_REGISTERED	954
ALTHING_DEFAULT_VISIBILITY_HIDDEN	954
ALTHING_MARBLES_LIMIT	954
ALTHING_STATUS_CLOSED	954
althing_send_alerts()	954
<i>send an alert message to every email address in \$emails</i>	
althing_get_emails()	954
<i>create a neat array with email-addresses from a (possibly messy) text</i>	
ALTHING_STATUS_OPENED	954
ALTHING_STATUS_FROZEN	954
ALTHING_REFERENCE	966
althing_show_form()	966
<i>show the form where the user/visitor can compose a new post</i>	
axis_get_properties()	1245
<i>construct a list of default properties for this theme</i>	
axis_demodata()	1244
<i>add demonstration data to the system</i>	
axis_install.php	1244
<i>/program/themes/axis/axis_install.php -- installer of the Axis Theme</i>	
axis.class.php	1243
<i>/program/themes/axis/axis.class.php - implements the Axis Theme</i>	
axis_install()	1245
<i>install the theme</i>	
axis_uninstall()	1246
<i>uninstall the theme</i>	
axis.php	1250
<i>/program/themes/axis/languages/nl/axis.php - translated messages for theme (Nederlands)</i>	
axis.php	1249
<i>/program/themes/axis/languages/en/axis.php - translated messages for theme (English)</i>	
axis_manifest.php	1248
<i>/program/themes/axis/axis_manifest.php - description of the axis theme</i>	
axis_upgrade()	1246
<i>upgrade the theme</i>	

althing.php	975
<i>/program/modules/althing/languages/nl/althing.php - translated messages for module (Dutch)</i>	
althing.php	974
<i>/program/modules/althing/languages/en/althing.php - translated messages for module (English)</i>	
althing_submit_message()	968
<i>store the new post, add relations and send mail to moderators (and maybe subscribers too)</i>	
althing_show_thankyou()	967
<i>display a thank-you message and the URL of the newly added post</i>	
althing_show_preview()	967
<i>show a preview of the post to the user, including references to parents etc.</i>	
althing_show_overview()	967
<i>display the althing information and all published posts OR a single post</i>	
althing_view()	969
<i>display the content of the althing linked to node \$node_id</i>	
althing_view_dialog_validate()	969
<i>validate the data entered by the user/visitor</i>	
althing_tabledefs.php	973
<i>/program/modules/althing/install/althing_tabledefs.php - data definition for module</i>	
althing_view_show_post()	970
<i>output a single post via \$theme</i>	
althing_view_get_posts()	970
<i>get all posts for this althing into an array for easy output</i>	
althing_view_get_dialogdef()	969
<i>construct a dialog definition for the add a post form</i>	
ALTHING_DEFAULT_VISIBILITY	954
althing_common.php	954
<i>/program/modules/althing/althing_common.php - code shared between admin and view</i>	
aggregator_view_get_modules()	935
<i>retrieve a list of modules suitable for aggregation keyed by module_id</i>	
aggregator_view_get_nodes()	935
<i>construct an array with the node records to aggregate</i>	
aggregator_view_get_config()	934
<i>retrieve the configuration information for this aggregator</i>	
aggregator_view()	933
<i>display the aggregated information from the nodes linked to this aggregator</i>	
aggregator_view.php	933
<i>/program/modules/aggregator/aggregator_view.php - interface to the view-part of the module</i>	
aggregator_view_get_node_from_db()	935
<i>retrieve an array with node records straight from the database</i>	
aggregator_view_get_node_from_tree()	935
<i>construct an array with node records using cached tree in current area</i>	
aggregator.php	939
<i>/program/modules/aggregator/languages/en/aggregator.php - translated messages for module (English)</i>	
aggregator_tabledefs.php	938
<i>/program/modules/aggregator/install/aggregator_tabledefs.php - data definition for module</i>	
aggregator_view_snapshots()	936
<i>construct a title, inline slideshow and readmore prompt for a snapshots page</i>	
aggregator_view_htmlpage()	936
<i>construct a title, summary and readmore prompt for an htmlpage page</i>	
aggregator_search()	932
<i>search this module's content in selected nodes for keywords in \$qwords</i>	

aggregator_search.php	932
<i>/program/modules/aggregator/aggregator_search.php - interface to the search-part of this module</i>	
aggregator_cron()	927
<i>routine that is called periodically by cron</i>	
aggregator_cron.php	927
<i>/program/modules/aggregator/aggregator_cron.php - interface to the cron-part of the aggregator module</i>	
aggregator_show_edit()	926
<i>present the user with a dialog to modify the aggregator that are connected to node \$node_id</i>	
aggregator_save()	925
<i>save the modified content data of this module linked to node \$node_id</i>	
aggregator_install.php	928
<i>/program/modules/aggregator/aggregator_install.php - installer of the aggregator module</i>	
aggregator_demodata()	928
<i>add demonstration data to the system</i>	
aggregator_manifest.php	931
<i>/program/modules/aggregator/aggregator_manifest.php - description of the aggregator module</i>	
aggregator_upgrade()	930
<i>upgrade the module</i>	
aggregator_uninstall()	929
<i>uninstall the module</i>	
aggregator_install()	929
<i>install the module</i>	
aggregator.php	940
<i>/program/modules/aggregator/languages/nl/aggregator.php - translated messages for module (Dutch)</i>	
althing_admin.php	942
<i>/program/modules/althing/althing_admin.php - management interface for althing-module</i>	
althing_show_edit()	948
<i>dispatcher for showing various dialog screens</i>	
althing_show_content()	948
<i>show the main content screen (in fact documentation only)</i>	
althing_save_viewlog()	947
<i>almost a dummy routine but it must exist to handle the lone [Cancel] button</i>	
althing_save_report()	947
<i>a dummy routine that pretends to "save" the report selections</i>	
althing_show_edit_configuration()	949
<i>show a simple edit screen for the althing configuration</i>	
althing_show_edit_moderation()	949
<i>present the user with a dialog for moderation of althing posts at node \$node_id</i>	
althing_show_edit_viewlog()	952
<i>show a list of moderation messages to the user</i>	
althing_show_edit_report()	952
althing_show_edit_moderation_post()	951
<i>show a single post and allow for moderation of this post</i>	
althing_show_edit_moderation_list()	951
<i>show a tabular overview of all posts in this althing</i>	
althing_save_moderation()	946
<i>validate and save the modified althing configuration linked to node \$node_id</i>	
althing_save_configuration()	946
<i>validate and save the modified althing configuration linked to node \$node_id</i>	
althing_get_dialogdef_configuration()	943

<i>construct a dialog definition for the main althing configuration</i>	943
<u>althing_disconnect()</u>	943
<i>disconnect this module from a node</i>	
<u>althing_connect()</u>	943
<i>connect this module to a node</i>	
<u>althing_config_dialog_validate()</u>	942
<i>validation of configuration dialog + rewriting the lists of email addresses as a side effect</i>	
<u>althing_get_dialogdef_moderation()</u>	944
<i>construct the moderation dialog</i>	
<u>althing_get_dialogdef_report()</u>	944
<i>construct the dialog for the reporting tool</i>	
<u>althing_save()</u>	945
<i>dispatcher for save routines</i>	
<u>althing_report_dialog_validate()</u>	945
<i>validate the report selection dialog, correct the list of posts if necessary</i>	
<u>althing_moderation_dialog_validate()</u>	945
<i>validation of moderation dialog</i>	
<u>althing_get_post_records()</u>	944
<i>retrieve the information about all posts in an array</i>	
<u>AreaManager::\$areas</u>	248
<u>AreaManager</u>	247
<i>Methods to access properties of an area</i>	
<u>AcIManager::\$area view enabled</u>	207
<u>AcIManager::\$a params save</u>	207
<u>AcIManager::\$area view a params</u>	207
<u>AcIManager::\$area view areas open</u>	207
<u>AcIManager::\$acl type</u>	207
<u>AcIManager::\$dialog</u>	208
<u>AcIManager::\$dialogdef</u>	208
<u>AcIManager::\$intro</u>	209
<u>AcIManager::\$header</u>	208
<u>AcIManager::\$dialogdef areas total</u>	208
<u>AcIManager::\$dialogdef areas</u>	208
<u>AcIManager::\$acl id</u>	206
<u>AcIManager</u>	202
<i>class for manipulating (edit+save) access control lists</i>	
<u>appropriate_legal_notices()</u>	144
<i>construct a link to appropriate legal notices as per GPLv3 section 5</i>	
<u>ACL_ROLE_PAGEMANAGER_SITEMASTER</u>	135
<u>ACL_ROLE_PAGEMANAGER_SECTIONMASTER</u>	135
<u>ACL_ROLE_PAGEMANAGER_PAGEMASTER</u>	135
<u>add_javascript_popup_function()</u>	176
<i>add javascript code that implements a popup to the header part of the page</i>	
<u>add_javascript_select_url_function()</u>	176
<i>add javascript code that implements a url selection (used in integration with CKEditor/FCKeditor)</i>	
<u>admin_show_login_and_exit()</u>	178
<i>show login dialog and exit</i>	
<u>admin_logout_and_exit()</u>	178
<i>logout the user and exit</i>	
<u>admin_login()</u>	177
<i>perform a step in the login procedure</i>	
<u>admin_continue_session()</u>	177

<i>continue the session from the previous call OR exit</i>	209
AclManager::\$output	209
AclManager::\$pagination_a_params	216
AclManager::get_permissions_nodes_in_area()	217
<i>retrieve an array with 0, 1 or more records with permissions from table 'acls_nodes'</i>	
AclManager::get_roles_intranet()	216
<i>construct an option list with roles for intranet access</i>	
AclManager::get_permissions_areas()	215
<i>retrieve an array with 0, 1 or more records with permissions from table 'acls_areas'</i>	
AclManager::get_permissions()	215
<i>retrieve an array with 0, 1 or more records with permissions from table 'acls'</i>	
AclManager::get_icon_area()	215
<i>construct a clickable icon to open/close this area</i>	
AclManager::get_roles_pagemanager()	217
<i>construct an option list with roles for pagemanager access</i>	
AclManager::save_data()	217
<i>save the changed data for the selected acl_type</i>	
AclManager::save_data_permissions()	218
<i>save the changed roles in the dialog to the corresponding tables 'acls'</i>	
AclManager::save_data_pagemanager()	217
<i>save the changed roles for pagemanager to the tables 'acls' and 'acls_areas' and 'acls_nodes'</i>	
AclManager::save_data_intranet()	217
<i>save the changed roles for intranet access to the tables 'acls' and 'acls_areas'</i>	
AclManager::save_data_admin()	217
<i>save changed job permissions to the database</i>	
AclManager::get_dialogdef_pagemanager()	214
<i>construct a dialog definition for pagemanager permissions</i>	
AclManager::get_dialogdef_intranet()	214
<i>construct an array with the intranet dialog information</i>	
AclManager::\$pagination_total	210
AclManager::\$pagination_offset	210
AclManager::\$pagination_limit	209
AclManager::\$pagination_enabled	209
AclManager::\$related_acls	210
AclManager::calc_areas_total()	211
<i>calculate the total number of items (site, areas, nodes) to show in dialog</i>	
AclManager::get_dialogdef_admin()	213
<i>construct an array with the admin dialog information</i>	
AclManager::enable_pagination()	213
<i>further initialise the AclManager and enable the dialog pagination feature</i>	
AclManager::enable_area_view()	212
<i>further initialise the AclManager and enable the area expand/collapse feature</i>	
AclManager::dialog_tableform()	211
<i>construct a form with a dialog in a table with 2 or 3 columns</i>	
ACL_ROLE_PAGEMANAGER_CONTENTMASTER	135
ACL_ROLE_PAGEMANAGER_AREAMASTER	135
ALERTMANAGER_CHORE_DELETE	26
ALERTMANAGER_CHORE_EDIT	26
ALERTMANAGER_CHORE_ADD	26
alertmanager.class.php	26
<i>/program/lib/alertmanager.class.php - manage alerts for changes in areas/nodes</i>	
ACL_TYPE_PAGEMANAGER	25
<i>acl for pagemanager permissions</i>	

ALERTMANAGER CHORE RULE ADD	26
ALERTMANAGER CHORE RULE DELETE	26
ALERTMANAGER CHORE VIEW	26
ALERTMANAGER CHORE SAVE	26
ALERTMANAGER CHORE RULE SAVE	26
ALERTMANAGER CHORE RULE EDIT	26
ACL_TYPE_MODULE	25
<i>acl for individual module permissions</i>	
ACL_TYPE_INTRANET	24
<i>acl for intranet permissions</i>	
aclmanager.class.php	24
<i>/program/lib/aclmanager.class.php - dealing with access control lists</i>	
accountmanagerlib.php	21
<i>/program/lib/accountmanagerlib.php - accountmanager (users and groups)</i>	
admin.php	18
<i>/program/languages/nl/admin.php - translated messages for /program/admin.php (Dutch)</i>	
admin.php	15
<i>/program/languages/en/admin.php - translated messages for /program/admin.php (English)</i>	
ACL_LEVEL_AREA	24
<i>limit available role options to pages, sections and areas (used in pagemanager permissions)</i>	
ACL_LEVEL_NONE	24
<i>limit available role options to 'none' and 'guru' (used in pagemanager permissions)</i>	
ACL_TYPE_ADMIN	24
<i>acl for administrator permissions</i>	
ACL_LEVEL_SITE	24
<i>no limit on available role options (used in pagemanager permissions)</i>	
ACL_LEVEL_SECTION	24
<i>limit available role options to pages and sections (used in pagemanager permissions)</i>	
ACL_LEVEL_PAGE	24
<i>limit available role options to pages (used in pagemanager permissions)</i>	
areamanager.class.php	27
<i>/program/lib/areamanager.class.php - taking care of area management</i>	
AREAMANAGER CHORE ADD	27
accesskey_from_string()	53
<i>return the ASCII-character that follows the first tilde in a string</i>	
ATTR_CLASS_VIEWONLY	52
ATTR_CLASS_ERROR	52
AREAMANAGER DIALOG RESET THEME	27
accesskey_tilde_to_underline()	53
<i>replace tilde+character with emphasised character to indicate accesskey</i>	
acceptable_new_password()	83
<i>check the new passwords satisfy password requirements</i>	
ACL_ROLE_NONE	135
ACL_ROLE_INTRANET_ACCESS	135
ACL_ROLE_GURU	135
authenticate_user()	84
<i>check the user's credentials in one of three ways</i>	
AREAMANAGER DIALOG EDIT THEME	27
AREAMANAGER DIALOG EDIT	27
AREAMANAGER CHORE RESET THEME	27
AREAMANAGER CHORE EDIT THEME	27
AREAMANAGER CHORE EDIT	27
AREAMANAGER CHORE DELETE	27

AREAMANAGER CHORE SAVE	27
AREAMANAGER CHORE SAVE_NEW	27
AREAMANAGER DIALOG DELETE	27
AREAMANAGER DIALOG ADD	27
AREAMANAGER CHORE VIEW	27
AREAMANAGER CHORE SET_DEFAULT	27
AclManager::set_action()	218
<i>further initialise the AclManager with the dialog action property</i>	
AclManager::set_dialog()	218
<i>further initialise the AclManager with the dialog identification</i>	
AdminSkin::\$icon_width	238
AdminSkin::\$knob_height	238
AdminSkin::\$icon_path	237
AdminSkin::\$icon_height	237
AdminSkin	237
<i>change the looks of the user interface</i>	
AdminSkin::\$knob_width	238
AdminSkin::\$name	238
AdminSkin::get_icon()	239
<i>return ready-to-use HTML-code for an anchor (to be used with an A-tag)</i>	
AdminSkin::\$text_only	239
AdminSkin::\$text_icons	239
AdminSkin::\$stylesheets	238
AdminOutput::set_suppress_output()	236
<i>manipulate output suppression</i>	
AdminOutput::set_helptopic()	236
<i>set the additional help topic to show when user clicks help button</i>	
AdminOutput::get_pagination()	234
<i>retrieve/construct a list of 0 or more clickable links to paginated screens</i>	
AdminOutput::get_navigation()	233
<i>construct a navigation bar for various jobs the user is allowed to do</i>	
AdminOutput::get_menu()	233
<i>get all lines in the menu DIV in a single properly indented string</i>	
AdminOutput::get_logo()	233
<i>construct an image tag with the W</i>	
AdminOutput::get_popups()	235
<i>construct javascript alerts for messages</i>	
AdminOutput::get_quickbottom()	235
<i>construct a list of quicklinks for bottom of page</i>	
AdminOutput::set_funnel_mode()	236
<i>manipulate the funnel mode</i>	
AdminOutput::send_output()	236
<i>send collected output to user's browser</i>	
AdminOutput::send_headers()	236
<i>send collected HTTP-headers to user's browser</i>	
AdminOutput::get_quicktop()	235
<i>construct a list of quicklinks for top of page, including logout link</i>	
AdminSkin::get_knob()	240
<i>return ready-to-use HTML-code for an anchor to be used in the navigation bar</i>	
AdminSkin::get_stylesheets()	240
<i>return the list of stylesheets associated with this skin</i>	
AlertManager::get_dialogdef_alert()	245
<i>construct a dialogue definition for adding/editing selected alert properties</i>	

AlertManager::dialog_validate_rule()	245
<i>validate add/edit rule dialog</i>	
AlertManager::dialog_validate_alert()	244
<i>validate add/edit alert dialog</i>	
AlertManager::a_param()	244
<i>shorthand for the anchor parameters that lead to the alert manager</i>	
AlertManager::get_dialogdef_rule()	245
<i>construct a dialogdef for adding/editing a rule for alert \$alert_id</i>	
AlertManager::run()	245
<i>entry point</i>	
AlertManager::tree_walk()	246
<i>walk the tree and add all nodes to an options array</i>	
AlertManager::show_rule()	246
<i>show a dialogue where a rule can be edited</i>	
AlertManager::show_parent_menu()	246
<i>allow the caller to use the menu area (or not)</i>	
AlertManager::show_alert()	245
<i>show a dialog to add a new or edit an existing alert</i>	
AlertManager::alert_save()	244
<i>store a new or modified alert in the database</i>	
AlertManager::alert_rule_save()	243
<i>store a new or modified alert rule in the database</i>	
AlertManager::\$output	241
AlertManager::\$intervals	241
AlertManager	240
<i>Methods to access properties of an alert</i>	
AdminSkin::is_text_only()	240
<i>is this skin a text-only skin?</i>	
AlertManager::\$show_parent_menu	241
AlertManager::alerts_overview()	242
<i>show link to 'add an alert' followed by a list of existing alerts</i>	
AlertManager::alert_rule_edit()	243
<i>display dialogue for a rule (new or existing)</i>	
AlertManager::alert_rule_delete()	243
<i>handle confirmation and actual delete of a rule</i>	
AlertManager::alert_edit()	242
<i>display dialogue for an alert (new or existing)</i>	
AlertManager::alert_delete()	242
<i>handle confirmation and actual delete of an alert</i>	
AdminOutput::get_lmth()	233
<i>proof of concept for braille-skin</i>	
AdminOutput::get_lines()	232
<i>get lines from an array in a single properly indented string</i>	
AdminOutput::\$html_head	224
AdminOutput::\$http_headers	224
AdminOutput::\$helptopic	223
AdminOutput::\$funnel_mode	223
AdminOutput::\$dtd	223
AdminOutput::\$menu	224
AdminOutput::\$messages_bottom	224
AdminOutput::\$skin	225
AdminOutput::\$pagination	225
AdminOutput::\$messages_top	225

AdminOutput::\$messages inline	224
AdminOutput::\$content	223
AdminOutput::\$breadcrumbs	223
AclManager::show_dialog()	219
<i>show the dialog where the selected Acl can be modified</i>	
AclManager::set_related_acls()	219
<i>further initialise the AclManager with related Acl's</i>	
AclManager::set_intro()	219
<i>further initialise the AclManager with the dialog introductory text</i>	
AclManager::set_header()	218
<i>further initialise the AclManager with the dialog header</i>	
AclManager::show_dialog_admin()	219
<i>display a tabular form for manipulating admin permissions</i>	
AclManager::show_dialog_intranet()	220
<i>display a tabular form for manipulating intranet permissions</i>	
AdminOutput	222
<i>conveniently collect output</i>	
AclManager::tree_build()	221
<i>build a tree of all nodes in an area</i>	
AclManager::show_tree_walk()	220
<i>add the specified node to dialogdef, optionally all subtrees, and subsequently all siblings</i>	
AclManager::show_dialog_pagemanager()	220
<i>display a tabular form for manipulating pagemanager permissions</i>	
AdminOutput::\$subtitle	225
AdminOutput::\$suppress_output	225
AdminOutput::get_address()	230
<i>return the reconstructed URL in a single (indented) line</i>	
AdminOutput::add_stylesheet()	230
<i>add a link to a stylesheet to the HTML head part of the document</i>	
AdminOutput::add_popup_top()	230
<i>add a message to the list of popup-messages at the TOP of the document</i>	
AdminOutput::add_popup_bottom()	230
<i>add a message to the list of popup-messages at the BOTTOM of the document</i>	
AdminOutput::get_bottomline()	230
<i>report basic performance indicators in a single line</i>	
AdminOutput::get_breadcrumbs()	231
<i>retrieve/construct a list of 0 or more clickable breadcrumbs</i>	
AdminOutput::get_html_head()	232
<i>get all lines in the HTML head section in a single properly indented string</i>	
AdminOutput::get_html()	232
<i>construct an output page in HTML</i>	
AdminOutput::get_div_messages()	231
<i>get a perhaps bulleted list of messages in a DIV</i>	
AdminOutput::get_content()	231
<i>get all lines in the content DIV in a single properly indented string</i>	
AdminOutput::add_pagination_item()	229
<i>add a link to screen of a paginated list to the existing list</i>	
AdminOutput::add_pagination()	228
<i>add a pagination navigation bar to the output</i>	
AdminOutput::add_content()	227
<i>add a line or array of lines to the content part of the document</i>	
AdminOutput::add_breadcrumb()	227
<i>add a breadcrumb to the breadcrumb trail</i>	

AdminOutput::\$title	226
AdminOutput::\$text_only	226
AdminOutput::add_html_header()	227
<i>add a header to the HTML head part of the document</i>	
AdminOutput::add_http_header()	227
<i>add an HTTP-header</i>	
AdminOutput::add_meta_http_equiv()	228
<i>add a line with http-equiv meta-information to the HTML head part of the document</i>	
AdminOutput::add_meta()	228
<i>add a line with meta-information to the HTML head part of the document</i>	
AdminOutput::add_message()	228
<i>add a message to the list of inline messages, part of the BODY of the document</i>	
AdminOutput::add_menu()	228
<i>add a line to the menu part of the document</i>	
admin.php	2
<i>/admin.php - the main entryptpoint for website maintenance</i>	

B

bbcode2html()	955
<i>convert valid BBCode to valid HTML</i>	
bg_manifest.php	516
<i>/program/languages/bg/bg_manifest.php - description of the Bulgarian translation</i>	
BY_PASSWORD	82
<i>this selects authentication via username+password in authenticate_user()</i>	
bbcode_callback()	956
<i>handle advanced BBCode search/replace specifically for encoding filenames</i>	
bbcode_callback_q()	957
<i>one-line wrapper to allow for fully qualified URLs</i>	
bbcode_valid_path()	958
<i>validate a filename for existence and access</i>	
bbcode_validate()	957
<i>validate the BBCode in the dialogdef item</i>	
bbcode_help()	957
<i>show a table illustrating the possible BBCode</i>	
BY_LAISSEZ_PASSER	82
<i>this selects authentication via username+laissez_passer in authenticate_user()</i>	
BY_EMAIL	82
<i>this selects authentication via username+email in authenticate_user()</i>	
BUTTON_EDIT	52
BUTTON_DONE	52
BUTTON_DELETE	52
BUTTON_GO	52
BUTTON_NO	52
BUTTON_YES	52
BUTTON_SAVE	52
BUTTON_OK	52
BUTTON_CANCEL	52

C

confab_common.php	987
<i>/program/modules/confab/confab_common.php - code shared between admin and view</i>	
confab_show_edit_report()	985
<i>present the user with a dialog for creating reports on confab conversations at node \$node_id</i>	
confab_show_edit_moderation_messages()	985
<i>create a list of all message in chronological order descending</i>	
confab_show_edit_moderation_list()	984
<i>show a button bar and list of messages in the selected conversation</i>	
CONFAB_ACTION_MESSAGE	987
CONFAB_ACTION_REMOVE	987
CONFAB_MAX_CONVERSATION_LIMIT	987
CONFAB_BRAILLE	987
CONFAB_ANONYMOUS_POSTFIX	987
confab_show_edit_moderation()	983
<i>present the user with one of two dialogs for moderation of confab posts at node \$node_id</i>	
confab_show_edit_configuration()	982
<i>show a simple edit screen for the confab configuration</i>	
confab_save()	979
<i>dispatcher for save routines</i>	
confab_get_dialogdef_report()	979
<i>construct the dialog for the reporting tool</i>	
confab_get_dialogdef_moderation()	979
<i>construct the moderation dialog</i>	
confab_get_dialogdef_configuration()	979
<i>construct a dialog definition for the main confab configuration</i>	
confab_save_configuration()	980
<i>validate and save the modified confab configuration linked to node \$node_id</i>	
confab_save_moderation()	981
<i>a dummy routine that pretends to "save" the moderation selections</i>	
confab_show_edit()	982
<i>dispatcher for showing various dialog screens</i>	
confab_show_content()	981
<i>show the main content screen (in fact documentation only)</i>	
confab_save_report()	981
<i>a dummy routine that pretends to "save" the report selections</i>	
CONFAB_MAX_PARTICIPANTS_LIMIT	987
CONFAB_MAX_TRIES_LIMIT	987
CONFAB_VISUAL_RB	987
CONFAB_VISUAL_BY	987
CONFAB_VISUAL_BW	987
CONFAB_VISUAL	987
confab_conversation_timeout()	987
<i>check conversations for timeout and hard limit and maybe create a new participant_id</i>	
confab_get_participants()	988
<i>retrieve all participants that joined the current conversation</i>	
confab_cron()	989
<i>routine that is called periodically by cron</i>	
confab_cron.php	989
<i>/program/modules/confab/confab_cron.php - interface to the cron-part of the confab module</i>	
confab_remote_addr()	988
<i>shorthand for remote IP-address</i>	
CONFAB_USER_STATUS_NONE	987
CONFAB_USER_STATUS_LEAVING	987

CONFAB_PERSONAL_YES	987
CONFAB_PERSONAL_REGISTERED	987
CONFAB_PERSONAL_NO	987
CONFAB_REFRESH_LIMIT	987
CONFAB_STATUS_CLOSED	987
CONFAB_USER_STATUS_JOINED	987
CONFAB_STATUS_REGISTERED	987
CONFAB_STATUS_OPENED	987
confab_get_conversations()	978
<i>retrieve all conversations in reverse chronological order</i>	
confab_disconnect()	978
<i>disconnect this module from a node</i>	
CrewWorkshop::calc_ranges()	834
constructor CrewWorkshop::CrewWorkshop()	834
CrewWorkshop::\$wid	834
CrewWorkshop::\$text	833
CrewWorkshop::cast_message()	834
CrewWorkshop::cast_userlist()	834
CrewWorkshop::disjoin()	834
CrewWorkshop::cast_user_leaves()	834
CrewWorkshop::cast_user_enters()	834
CrewWorkshop::\$origin	833
CrewWorkshop::\$name	833
CrewServer::initialise()	833
CrewServer::get_tv_sec()	833
CrewServer::frame_encode()	832
CrewServer::lookup_client_cid()	833
CrewServer::run()	833
CrewWorkshop::\$clients	833
CrewWorkshop::\$attr	833
CrewWorkshop	833
CrewWorkshop::get_next_attr()	834
CrewWorkshop::get_userlist()	834
crew.php	897
<i>/program/modules/crew/languages/vi/crew.php</i>	
confab.php	896
<i>/program/modules/confab/languages/vi/confab.php</i>	
cornelia.php	882
<i>/program/themes/cornelia/languages/ur/cornelia.php</i>	
crew.php	875
<i>/program/modules/crew/languages/ur/crew.php</i>	
cornelia.php	904
<i>/program/themes/cornelia/languages/vi/cornelia.php</i>	
check_add_reply_harvest()	971
<i>check for user action indicating wish to add a post, reply to a post or harvest posts</i>	
confab_connect()	978
<i>connect this module to a node</i>	
confab_config_dialog_validate()	977
<i>validation of configuration dialog + rewriting passcode (no trailing/leading spaces)</i>	
confab_admin.php	977
<i>/program/modules/confab/confab_admin.php - management interface for confab-module</i>	
cornelia.php	866
<i>/program/themes/cornelia/languages/tr/cornelia.php</i>	

cornelia.php	854
<i>/program/themes/cornelia/languages/sv/cornelia.php</i>	
CrewWorkshop::send()	835
CrewWorkshop::process_diff()	835
CrewWorkshop::join()	835
CrewWorkshop::send_text_full()	835
CrewWorkshop::send_userlist()	835
crew.php	847
<i>/program/modules/crew/languages/sv/crew.php</i>	
constructor FCKeditor::FCKeditor()	837
<i>Main Constructor.</i>	
<i>Refer to the <code>_samples/php</code> directory for examples.</i>	
CrewWorkshop::send_user_info()	835
confab_install.php	990
<i>/program/modules/confab/confab_install.php - installer of the confab module</i>	
confab_demodata()	990
<i>add demonstration data to the system</i>	
crew_screen()	1024
<i>construct triple-indirect edit screen in pop-up</i>	
CREW_MAX_DOCUMENT_SIZE	1024
crew_view.php	1024
<i>/program/modules/crew/crew_view.php - interface to the view-part of the crew module</i>	
crew_search()	1022
<i>search this module's content in selected nodes for keywords in \$qwords</i>	
crew_view()	1025
<i>display the content of the workshop linked to node \$node_id</i>	
crew_view_dialogdef()	1025
<i>construct an option to select a skin and start an Edit session</i>	
crew_view_show_edit()	1026
<i>show the (visually almost) empty page and load or continue with the JS popup window</i>	
crew_view_save_document()	1026
<i>save the POST'ed version of the document to the workshop database</i>	
crew_view_get_workshop_data()	1026
<i>retrieve the current version of the document + other workshop data</i>	
crew_search.php	1022
<i>/program/modules/crew/crew_search.php - interface to the search-part of this module</i>	
crew_manifest.php	1021
<i>/program/modules/crew/crew_manifest.php - description of the crew module</i>	
crew_cron()	1017
<i>routine that is called periodically by cron</i>	
crew_cron.php	1017
<i>/program/modules/crew/crew_cron.php - interface to the cron-part of the workshop (CREW) module</i>	
crew_show_edit()	1015
<i>present the user with a dialog to modify the workshop that is connected to node \$node_id</i>	
crew_install.php	1018
<i>/program/modules/crew/crew_install.php - installer of the crew module</i>	
crew_demodata()	1018
<i>add demonstration data to the system</i>	
crew_upgrade()	1020
<i>upgrade the module</i>	
crew_uninstall()	1019
<i>uninstall the module</i>	

crew_install()	1019
<i>install the module</i>	
crew_view_show_view()	1027
<i>display the current version of the document and maybe an Edit button</i>	
crew_tabledefs.php	1029
<i>/program/modules/crew/install/crew_tabledefs.php - data definition for module</i>	
constructor ThemeRosalina::ThemeRosalina()	1284
<i>construct a ThemeRosalina object</i>	
constructor ThemeCornelia::ThemeCornelia()	1261
cornelia.php	1260
<i>/program/themes/cornelia/languages/nl/cornelia.php - translated messages for theme (Nederlands)</i>	
cornelia.php	1259
<i>/program/themes/cornelia/languages/en/cornelia.php - translated messages for theme (English)</i>	
constructor ThemeRuta::ThemeRuta()	1298
<i>construct a ThemeRuta object</i>	
constructor ThemeSchoolyard::ThemeSchoolyard()	1314
<i>construct a ThemeSchoolyard object</i>	
CREDITS	1387
CHANGES	1343
constructor ThemeSophia::ThemeSophia()	1323
cornelia_manifest.php	1258
<i>/program/themes/cornelia/cornelia_manifest.php - description of the cornelia theme</i>	
cornelia_upgrade()	1256
<i>upgrade the theme</i>	
constructor SnapshotViewer::SnapshotViewer()	1235
<i>the constructor only stores relevant data for future use</i>	
crew.php	1031
<i>/program/modules/crew/languages/nl/crew.php - translated messages for module (Dutch)</i>	
crew.php	1030
<i>/program/modules/crew/languages/en/crew.php - translated messages for module (English)</i>	
constructor SnapshotViewerInline::SnapshotViewerInline()	1239
<i>the constructor only stores relevant data for future use</i>	
cornelia_install.php	1254
<i>/program/themes/cornelia/cornelia_install.php -- installer for the cornelia theme</i>	
cornelia_uninstall()	1256
<i>uninstall the theme</i>	
cornelia_install()	1255
<i>install the theme</i>	
cornelia_demodata()	1254
<i>add demonstration data to the system</i>	
crew_save()	1014
<i>save the modified content data of this module linked to node \$node_id</i>	
crew_get_dialogdef()	1013
<i>construct a dialog definition for the workshop configuration</i>	
confab_braille_show_talk()	998
<i>show a dialog where a user can write a message or can navigate to another confab-screen</i>	
confab_braille_show_main()	998
<i>construct the main/messages screen for the braille interface</i>	
confab_braille_show_all()	997
<i>send all messages to the participant; update last_seen_id</i>	
confab_braille_page_headers()	997

<i>send various headers attempting to defeat caching</i>	998
<u>confab_braille_show_users()</u>	998
<i>send a list of users to the participant</i>	
<u>confab_join()</u>	999
<i>handle the complete procedure to join the conversation</i>	
<u>confab_join_dialog_validate()</u>	1000
<i>validate the data entered by the user/visitor</i>	
<u>confab_join_dialog_show()</u>	1000
<i>output the join dialog including the form and header/introduction/footer</i>	
<u>confab_join_conversation()</u>	999
<i>do housekeeping and setup the opening screen with requested interface</i>	
<u>confab_braille_dispatcher()</u>	996
<i>dispatcher for braille interface</i>	
<u>CONFAB REFERENCE</u>	996
<u>confab_upgrade()</u>	992
<i>upgrade the module</i>	
<u>confab_uninstall()</u>	991
<i>uninstall the module</i>	
<u>confab_install()</u>	991
<i>install the module</i>	
<u>confab_manifest.php</u>	993
<i>/program/modules/confab/confab_manifest.php - description of the confab module</i>	
<u>confab_search.php</u>	994
<i>/program/modules/confab/confab_search.php - interface to the search-part of this module</i>	
<u>CONFAB DIRNAME</u>	996
<u>confab_view.php</u>	996
<i>/program/modules/confab/confab_view.php - interface to the view-part of the confab module</i>	
<u>confab_search()</u>	994
<i>search this module's content in selected nodes for keywords in \$qwords</i>	
<u>confab_join_get_dialogdef()</u>	1001
<i>construct a dialog definition for the join screen</i>	
<u>confab_leave_conversation()</u>	1002
<i>end the conversation for \$participant (at their own request or forced)</i>	
<u>CREW ACL ROLE READONLY</u>	1012
<u>crew_admin.php</u>	1012
<i>/program/modules/crew/crew_admin.php - management interface for crew-module</i>	
<u>confab.php</u>	1010
<i>/program/modules/confab/languages/nl/confab.php - translated messages for module (Dutch)</i>	
<u>confab.php</u>	1009
<i>/program/modules/confab/languages/en/confab.php - translated messages for module (English)</i>	
<u>CREW ACL ROLE READWRITE</u>	1012
<u>CREW PERMISSION READ</u>	1012
<u>crew_disconnect()</u>	1013
<i>disconnect this module from a node</i>	
<u>crew_connect()</u>	1012
<i>connect this module to a node</i>	
<u>CREW PERMISSION WRITE</u>	1012
<u>confab_tabledefs.php</u>	1008
<i>/program/modules/confab/install/confab_tabledefs.php - data definition for module</i>	
<u>confab_visual_update()</u>	1006
<i>send an update to the client in response to an XMLHttpRequest</i>	
<u>confab_view_calc_participants()</u>	1003

<i>calculate the number of currently joined participants in the current conversation</i>	
<u>confab_view()</u>	1002
<i>display the content of the confab linked to node \$node_id</i>	
<u>confab_save_message()</u>	1002
<i>add non-empty message to database</i>	
<u>confab_view_delete_stale_participants()</u>	1003
<i>check for stale participants and remove corresponding records</i>	
<u>confab_view_get_messages()</u>	1004
<i>retrieve all messages in the conversation since \$last_message_id</i>	
<u>confab_visual_show_main()</u>	1005
<i>construct the main screen for the Visual interfae</i>	
<u>confab_visual_dispatcher()</u>	1005
<i>dispatcher for visual interface</i>	
<u>confab_view_participants_timeout()</u>	1004
<i>check for idle participants and timed out participants</i>	
<u>CrewServer::find_workshop()</u>	832
<u>CrewServer::dump_overview()</u>	832
<u>ConfigAssistant::show_dialog()</u>	265
<i>add a complete dialog to the content area of the output</i>	
<u>ConfigAssistant::save_data()</u>	265
<i>save the modified configuration parameters to the database</i>	
<u>ConfigAssistant::get_options_from_extra()</u>	265
<u>ConfigAssistant::get_extra()</u>	264
<i>construct an array based on name=value pairs in an 'extra' field</i>	
<u>constructor DatabaseMysql::DatabaseMysql()</u>	269
<i>initialise query counter and other variables, store the table prefix</i>	
<u>constructor DatabaseMysqli::DatabaseMysqli()</u>	283
<i>initialise query counter and other variables, store the table prefix</i>	
<u>constructor Email::Email()</u>	300
<i>constructor resets all variables to a known (default) state</i>	
<u>constructor DatabaseMysqlResult::DatabaseMysqlResult()</u>	296
<i>constructor</i>	
<u>constructor DatabaseMysqliResult::DatabaseMysqliResult()</u>	294
<i>constructor</i>	
<u>ConfigAssistant::get_dialogdef()</u>	264
<i>construct an array with the dialog information</i>	
<u>constructor ConfigAssistant::ConfigAssistant()</u>	264
<i>constructor for the configuration assistant</i>	
<u>ConfigAssistant::\$keyfield</u>	263
<u>ConfigAssistant::\$fields</u>	262
<u>ConfigAssistant::\$dialogdef_hidden</u>	262
<u>ConfigAssistant::\$language_domain</u>	263
<u>ConfigAssistant::\$prefix</u>	263
<u>ConfigAssistant::\$where</u>	264
<u>ConfigAssistant::\$table</u>	263
<u>ConfigAssistant::\$records</u>	263
<u>constructor FileManager::FileManager()</u>	317
<i>construct a FileManager object (called from /program/main_admin.php)</i>	
<u>constructor GroupManager::GroupManager()</u>	339
<i>construct a GroupManager object</i>	
<u>crew.php</u>	576
<i>/program/modules/crew/languages/el/crew.php</i>	
<u>confab.php</u>	575

/program/modules/confab/languages/el/confab.php	561
cornelia.php	561
/program/themes/cornelia/languages/de/cornelia.php	554
crew.php	554
/program/modules/crew/languages/de/crew.php	594
crew.php	594
/program/modules/crew/languages/es/crew.php	598
cornelia.php	598
/program/themes/cornelia/languages/es/cornelia.php	663
confab.php	649
/program/modules/confab/languages/fry/confab.php	642
cornelia.php	642
/program/themes/cornelia/languages/fr/cornelia.php	528
crew.php	521
/program/modules/crew/languages/fr/crew.php	521
crew.php	521
/program/modules/crew/languages/bg/crew.php	396
constructor Theme::Theme()	357
construct a Theme object	
constructor PageManager::PageManager()	350
construct a PageManager object (called from /program/main_admin.php)	
constructor Language::Language()	411
constructor	
constructor TranslateTool::TranslateTool()	427
construct a TranslateTool object	
constructor Useraccount::Useraccount()	472
get pertinent user information in core	
constructor InstallWizard::InstallWizard()	449
constructor	
constructor Zip::Zip()	432
constructor initialises all variables	
constructor UserManager::UserManager()	262
construct a UserManager object	
ConfigAssistant::\$dialogdef	258
ConfigAssistant	
class for editing standard configuration tables	
CAPACITY_CUSTOM7	144
CAPACITY_CUSTOM6	144
CAPACITY_CUSTOM5	144
CAPACITY_CUSTOM4	144
CAPACITY_CUSTOM8	144
CAPACITY_CUSTOM9	144
CAPACITY_NEXT_AVAILABLE	144
CAPACITY_MEMBER	144
CAPACITY_EDITOR	144
CAPACITY_CUSTOM3	144
CAPACITY_CUSTOM2	144
configurationmanagerlib.php	29
/program/lib/configurationmanagerlib.php - configurationmanager	
configassistant.class.php	28
/program/lib/configassistant.class.php - dealing with lists of configuration parameters	

cron.php	7
<i>/cron.php - the main entrypoint for processing cron jobs</i>	
CHORE_SAVE	29
CHORE_SESSION_DELETE	113
CAPACITY_CUSTOM1	144
CAPACITY_CHAIR	144
CHORE_SESSION_VIEW	113
CAPACITY_NONE	144
<i>The constants CAPACITY_* are used for group memberships (see accountmanagerlib.php).</i>	
CAPACITY_PRINCIPAL	144
calculate_node_id()	190
<i>calculate and validate the node_id to display</i>	
calculate_default_page()	190
<i>try to find a default page within a subtree of pages and sections</i>	
calculate_area()	189
<i>try to retrieve a valid area record based on values of requested area and requested node</i>	
cron_send_queued_alerts()	147
<i>send pending messages/alerts</i>	
constructor AclManager::AclManager()	210
<i>constructor for the AclManager</i>	
constructor AdminOutput::AdminOutput()	226
<i>constructor</i>	
constructor AreaManager::AreaManager()	248
<i>construct an AreaManager object</i>	
constructor AlertManager::AlertManager()	242
<i>construct an AlertManager object</i>	
constructor AdminSkin::AdminSkin()	239
<i>construct an AdminSkin object (called from AdminOutput)</i>	
convert_to_type()	146
<i>convert a string to another type (bool, int, etc.)</i>	
capacity_name()	146
<i>translate a numeric capacity code to a readable name</i>	
CAPACITY_PUPIL	144
CAPACITY_PUBLISHER	144
CAPACITY_PROJECTLEAD	144
CAPACITY_SECRETARY	144
CAPACITY_TEACHER	144
calc_user_related_acls()	146
<i>calculate an array with acls related to user \$user_id via group memberships</i>	
calculate_uri_shortcuts()	145
<i>try to eliminate the scheme and authority from the two main uri's</i>	
CAPACITY_TREASURER	144
crew.php	664
<i>/program/modules/crew/languages/fry/crew.php</i>	
cornelia.php	671
<i>/program/themes/cornelia/languages/fry/cornelia.php</i>	
CrewClient::\$state	829
CrewClient::\$socket	829
CrewClient::\$server	829
CrewClient::\$remote_port	829
CrewClient::\$wid	829
CrewClient::\$workshop	829
CrewClient::frame_decode()	829

CrewClient::frame_available()	829
constructor CrewClient::CrewClient()	829
CrewClient::\$remote_address	829
CrewClient::\$range	829
CrewClient::\$date	828
CrewClient::\$cid	828
CrewClient::\$buf_out	828
CrewClient::\$headers	828
CrewClient::\$local_address	828
CrewClient::\$nick	829
CrewClient::\$name	828
CrewClient::\$local_port	828
CrewClient::process_request()	830
CrewClient::recv()	830
CrewServer::\$sockets	831
CrewServer::\$server_socket	831
CrewServer::\$server_run_flag	831
CrewServer::\$server_port	831
CrewServer::\$wids	832
CrewServer::\$workshops	832
CrewServer::disconnect_socket()	832
CrewServer::disconnect_client()	832
constructor CrewServer::CrewServer()	832
CrewServer::\$server_address	831
CrewServer::\$mark_time	831
CrewClient::valid_handshake()	830
CrewClient::valid_authentication()	830
CrewClient::send()	830
CrewServer	831
CrewServer::\$cids	831
CrewServer::\$mark_next	831
CrewServer::\$dirty_sockets	831
CrewServer::\$clients	831
CrewClient::\$buf_in	828
CrewClient::\$authenticated	828
CKEditor	822
<i>brief CKEditor class that can be used to create editor instances in PHP pages on server side.</i>	
CREW_SERVER_VERSION	820
CREW_SERVER_NAME	820
CREW_SERVER_DATE	820
CKEditor::\$basePath	822
<i>URL to the %CKEditor installation directory (absolute or relative to document root). If not set, CKEditor will try to guess it's path.</i>	
CKEditor::\$config	822
<i>An array that holds the global %CKEditor configuration. For the list of available options, see http://docs.cksource.com/ckeditor_api/symbols/CKEDITOR.config.html</i>	
CKEditor::\$textareaAttributes	823
<i>An array with textarea attributes.</i>	
CKEditor::\$returnOutput	823
<i>Boolean variable indicating whether created code should be printed out or returned by a function.</i>	

CKEditor::\$initialized	823
<i>A boolean variable indicating whether CKEditor has been initialized.</i>	
crewserver.php	820
ckeditor.php	815
crew.php	727
<i>/program/modules/crew/languages/pap/crew.php</i>	
cornelia.php	712
<i>/program/themes/cornelia/languages/it/cornelia.php</i>	
crew.php	705
<i>/program/modules/crew/languages/it/crew.php</i>	
cornelia.php	734
<i>/program/themes/cornelia/languages/pap/cornelia.php</i>	
crew.php	760
<i>/program/modules/crew/languages/pt/crew.php</i>	
cornelia.php	797
<i>/program/themes/cornelia/languages/ru/cornelia.php</i>	
crew.php	790
<i>/program/modules/crew/languages/ru/crew.php</i>	
cornelia.php	767
<i>/program/themes/cornelia/languages/pt/cornelia.php</i>	
CKEditor::\$timestamp	823
<i>A string indicating the creation date of %CKEditor.</i>	
CKEditor::\$version	823
<i>The version of %CKEditor.</i>	
CKEditor::replace()	827
<i>Replaces a <code><textarea></code> with a %CKEditor instance.</i>	
CKEditor::jsEncode()	826
<i>This little function provides a basic JSON support.</i>	
<i>private</i>	
CKEditor::init()	826
<i>Initializes CKEditor (executed only once).</i>	
CKEditor::editor()	826
<i>Creates a %CKEditor instance.</i>	
<i>In incompatible browsers %CKEditor will downgrade to plain HTML <code><textarea></code> element.</i>	
CKEditor::replaceAll()	827
<i>Replace all <code><textarea></code> elements available in the document with editor instances.</i>	
CKEditor::returnGlobalEvents()	827
<i>Return global event handlers.</i>	
CrewClient::\$attr	828
CrewClient	828
CKEditor::script()	827
<i>Prints javascript code.</i>	
<i>private</i>	
CKEditor::configSettings()	826
<i>Returns the configuration array (global and instance specific settings are merged into one array).</i>	
<i>private</i>	
CKEditor::clearGlobalEventHandlers()	825
<i>Clear registered global event handlers.</i>	
<i>Note: this function will have no effect if the event handler has been already printed/returned.</i>	
CKEditor::\$ timestamp	824

<i>A constant string unique for each release of %CKEditor.</i>	
CKEditor::\$ globalEvents	824
<i>An array that holds global event listeners.</i>	
CKEditor::\$ events	824
<i>An array that holds event listeners.</i>	
constructor CKEditor::CKEditor()	824
<i>Main Constructor.</i>	
CKEditor::addEventHandler()	824
<i>Adds event listener.</i>	
<i>Events are fired by %CKEditor in various situations.</i>	
CKEditor::clearEventHandlers()	825
<i>Clear registered event handlers.</i>	
<i>Note: this function will have no effect on already created editor instances.</i>	
CKEditor::ckeditorPath()	825
<i>Return path to ckeditor.js.</i>	
CKEditor::addGlobalEventHandler()	825
<i>Adds global event listener.</i>	
config-example.php	3
<i>/config-example.php - example of the main configuration file</i>	

D

DatabaseMysqli::last_insert_id()	291
<i>retrieve the most recent automatically inserted id ('auto_increment')</i>	
DatabaseMysqli::mysqli_utf8mb3()	291
<i>message string to contain only 3-byte UTF8-sequences</i>	
DatabaseMysqli::mysqli_utf8_support()	291
<i>determine the level of UTF-8 support based on MySQL-server version</i>	
DatabaseMysqli::query()	292
<i>execute a select query and return a result set</i>	
DatabaseMysqli::exec()	290
<i>execute an action query and return the number of affected rows</i>	
DatabaseMysqli::escape()	290
<i>escape special characters in string</i>	
DatabaseMysqli::create_table_sql()	288
<i>create the MySQL-specific SQL statement to create a table via a generic table definition</i>	
DatabaseMysqli::drop_table()	289
<i>unconditionally drop the specified table</i>	
DatabaseMysqli::dump()	289
<i>make a text dump of our tables in the database suitable for backup purposes</i>	
DatabaseMysqli::table_exists()	293
<i>see if the named table exists</i>	
DatabaseMysqliResult	293
<i>MySQL database result</i>	
DatabaseMysqliResult::close()	294
<i>free the memory associated with the result set</i>	
DatabaseMysqliResult::fetch_all()	294
<i>fetch all rows as a 0-based array of 0-based enumerated arrays</i>	
DatabaseMysqliResult::fetch_all_assoc()	295
<i>fetch all rows as an array (0-based or keyed) of associative arrays</i>	
DatabaseMysqliResult::fetch_row()	295
<i>fetch the next result row as a 0-based enumerated array</i>	

DatabaseMysqliResult::\$result	294
DatabaseMysqliResult::\$num_rows	294
DatabaseMysqliResult::\$db_link	293
DatabaseMysqliResult::\$errno	293
DatabaseMysqliResult::\$error	294
DatabaseMysqli::create_table()	288
<i>create a table via a generic (non-MySQL-specific) table definition</i>	
DatabaseMysqli::connect()	287
<i>connect to the database server and open the database</i>	
DatabaseMysqli::\$db_server	281
DatabaseMysqli::\$db_type	281
DatabaseMysqli::\$db_username	281
DatabaseMysqli::\$db_version	281
DatabaseMysqli::\$db_password	280
DatabaseMysqli::\$db_name	280
DatabaseMysqli::\$charset	280
DatabaseMysqli::\$collation	280
DatabaseMysqli::\$db_link	280
DatabaseMysqli::\$debug	281
DatabaseMysqli::\$engine	282
DatabaseMysqli::check_engine()	283
<i>determine the database engine (and charset and collation) to use for new tables</i>	
DatabaseMysqli::close()	284
<i>close the connection to the database</i>	
DatabaseMysqli::column_definition()	285
<i>convert a fielddef array to a MySQL specific column definition</i>	
DatabaseMysqli::concat()	286
<i>helper function for string concatenation in sql statements</i>	
DatabaseMysqli::\$utf8_support	283
DatabaseMysqli::\$query_counter	282
DatabaseMysqli::\$errno	282
DatabaseMysqli::\$error	282
DatabaseMysqli::\$prefix	282
DatabaseMysqliResult::fetch_row_assoc()	295
<i>fetch the next result row as a associative array</i>	
DatabaseMysqliResult	295
<i>MySQL database result</i>	
demodata.php	617
<i>/program/install/languages/fi/demodata.php</i>	
demodata.php	635
<i>/program/install/languages/fr/demodata.php</i>	
demodata.php	655
<i>/program/install/languages/fry/demodata.php</i>	
demodata.php	677
<i>/program/install/languages/hi/demodata.php</i>	
demodata.php	604
<i>/program/install/languages/fa/demodata.php</i>	
demodata.php	587
<i>/program/install/languages/es/demodata.php</i>	
demodata.php	547
<i>/program/install/languages/de/demodata.php</i>	
de_manifest.php	550
<i>/program/languages/de/de_manifest.php - description of the German translation</i>	

demodata.php	567
/program/install/languages/el/demodata.php	
demodata.php	685
/program/install/languages/hu/demodata.php	
demodata.php	698
/program/install/languages/it/demodata.php	
demodata.php	840
/program/install/languages/sv/demodata.php	
demodata.php	868
/program/install/languages/ur/demodata.php	
demodata.php	888
/program/install/languages/vi/demodata.php	
demodata.php	910
/program/install/languages/zh/demodata.php	
dump_buffer()	820
demodata.php	783
/program/install/languages/ru/demodata.php	
demodata.php	720
/program/install/languages/pap/demodata.php	
demodata.php	740
/program/install/languages/pl/demodata.php	
demodata.php	753
/program/install/languages/pt/demodata.php	
da_manifest.php	537
/program/languages/da/da_manifest.php - description of the Danish translation	
demodata.php	534
/program/install/languages/da/demodata.php	
DatabaseMysqlResult::fetch_all()	297
fetch all rows as a 0-based array of 0-based enumerated arrays	
DatabaseMysqlResult::fetch_all_assoc()	297
fetch all rows as an array (0-based or keyed) of associative arrays	
DatabaseMysqlResult::fetch_row()	297
fetch the next result row as a 0-based enumerated array	
DatabaseMysqlResult::fetch_row_assoc()	297
fetch the next result row as a associative array	
DatabaseMysqlResult::close()	296
free the memory associated with the result set	
DatabaseMysqlResult::\$result	296
DatabaseMysqlResult::\$errno	295
DatabaseMysqlResult::\$error	296
DatabaseMysqlResult::\$num_rows	296
demodata.php	458
/program/install/demodata.php - code to install the main demodata	
demodata()	458
insert basic demonstration data; the foundation for the module/theme demonstration data	
demodata.php	465
/program/install/languages/nl/demodata.php - translated messages for	
/program/install/demodata.php (Dutch)	
demodata.php	500
/program/install/languages/ar/demodata.php	
demodata.php	513
/program/install/languages/bg/demodata.php	
demodata.php	463

/program/install/languages/en/demodata.php - translated messages for	
/program/install/demodata.php (English)	
demodata_users_groups()	460
<i>create a handful of users/groups/capacities/acls</i>	
demodata_alerts()	459
<i>create a few alerts</i>	
demodata_areas()	459
<i>create three areas + themes</i>	
demodata_sections_pages()	460
<i>create a few sections and pages</i>	
DatabaseMysqli	279
<i>MySQL database</i>	
DatabaseMysqli::table_exists()	279
<i>see if the named table exists</i>	
dbsession_read()	46
<i>read the (serialised) session data from the database</i>	
dbsession_remove_obsolete_sessions()	46
<i>workhorse for removing obsolete sessions from the database</i>	
dbsession_setup()	47
<i>setup database based handlers for session management</i>	
dbsession_write()	48
<i>write the (serialised) data to the database</i>	
dbsession_open()	46
<i>'open' a session</i>	
dbsession_get_session_id()	45
<i>retrieve the session_id (pkey) that corresponds with session_key</i>	
dbsession_exists()	44
<i>check to see if \$session_key exists in the session table</i>	
dbsession_expire()	45
<i>remove all sessions that were created more than \$max_life seconds ago</i>	
dbsession_garbage_collection()	45
<i>remove all sessions that are last accessed more than \$time_out seconds ago, maybe even more</i>	
dialoglib.php	49
<i>/program/lib/dialoglib.php - useful functions for manipulating dialogs</i>	
dialog_buttondef()	54
<i>shortcut for generating a dialogdef for a button</i>	
dialog_get_widget_listbox()	57
<i>construct a listbox</i>	
dialog_get_widget_radiocheckbox()	58
<i>construct a checkbox or 1 or more radiobuttons</i>	
dialog_get_widget_richtextinput()	59
<i>construct an input field using the user's preferred editor</i>	
dialog_get_widget_submit()	60
<i>construct a submit button</i>	
dialog_get_widget_file()	56
<i>construct an input field for file upload</i>	
dialog_get_widget()	55
<i>construct an actual HTML input widget for dialog input element</i>	
dialog_csrf_token()	54
<i>shortcut for generating a dialogdef for a csrf_token</i>	
dialog_get_class()	54
<i>construct a space-delimited list of classes that apply to this item</i>	

dialog_get_label()	55
<i>construct a label for a dialog input element</i>	
dbsession_destroy()	44
<i>remove a session record from the sessions table (it should still exist)</i>	
dbsession_create()	43
<i>create a new session in the session table, return the unique sessionkey</i>	
db_errormessage()	33
<i>retrieve the latest database error from \$DB</i>	
db_escape_and_quote()	33
<i>conditionally quote and escape values for use with a database table</i>	
db_insert_into()	34
<i>execute the necessary SQL-code for an INSERT INTO statement</i>	
db_insert_into_and_get_id()	34
<i>execute the necessary SQL-code for an INSERT INTO statement and return the last_insert_id</i>	
db_delete_sql()	33
<i>generate SQL to delete zero or more rows in a table</i>	
db_delete()	32
<i>delete zero or more rows in a table</i>	
databaselib.php	31
<i>/program/lib/database/databaselib.php - database factory and database access routines</i>	
database_factory()	31
<i>manufacture a database object</i>	
db_bool_is()	32
<i>check boolean field in a database-independent way</i>	
db_insert_into_sql()	35
<i>generate the necessary SQL-code for an INSERT INTO statement</i>	
db_last_insert_id()	35
<i>wrapper for DB->last_insert_id()</i>	
db_where_clause()	39
<i>construct a where clause from string/array, including the word WHERE</i>	
dbsessionlib.php	42
<i>/program/lib/dbsessionlib.php - functions to keep PHP-sessions in the database</i>	
dbsession_close()	43
<i>'close' a session that was opened with dbsession_open() before</i>	
db_update_sql()	38
<i>generate sql to update one or more fields in a table</i>	
db_update()	38
<i>update one or more fields in a table</i>	
db_select_all_records()	36
<i>fetch all selected records from the database in one array</i>	
db_select_single_record()	36
<i>fetch a single record from the database</i>	
db_select_sql()	37
<i>generate the necessary SQL-code for a simple SELECT statement</i>	
dialog_get_widget_textinput()	61
<i>construct an input field, usually for text input OR a textarea for multiline input</i>	
dialog_quickform()	62
<i>construct a generic form with a dialog</i>	
DatabaseMysql::check_engine()	270
<i>determine the database engine (and charset and collation) to use for new tables</i>	
DatabaseMysql::close()	271
<i>close the connection to the database</i>	
DatabaseMysql::column_definition()	271

convert a fielddef array to a MySQL specific column definition	272
DatabaseMysql::concat()	272
helper function for string concatenation in sql statements	
DatabaseMysql::\$utf8_support	269
DatabaseMysql::\$query_counter	269
DatabaseMysql::\$errno	268
DatabaseMysql::\$error	269
DatabaseMysql::\$prefix	269
DatabaseMysql::connect()	274
connect to the database server and open the database	
DatabaseMysql::create_table()	274
create a table via a generic (non-MySQL-specific) table definition	
DatabaseMysql::last_insert_id()	277
retrieve the most recent automatically inserted id ('auto_increment')	
DatabaseMysql::mysql_utf8mb3()	277
massage string to contain only 3-byte UTF8-sequences	
DatabaseMysql::mysql_utf8_support()	278
determine the level of UTF-8 support based on MySQL-server version	
DatabaseMysql::query()	278
execute a select query and return a result set	
DatabaseMysql::exec()	277
execute an action query and return the number of affected rows	
DatabaseMysql::escape()	276
escape special characters in string	
DatabaseMysql::create_table_sql()	275
create the MySQL-specific SQL statement to create a table via a generic table definition	
DatabaseMysql::drop_table()	275
unconditionally drop the specified table	
DatabaseMysql::dump()	275
make a text dump of our tables in the database suitable for backup purposes	
DatabaseMysql::\$engine	268
DatabaseMysql::\$debug	268
DIALOG_NODE_EDIT_ADVANCED	100
DIALOG_NODE_EDIT_CONTENT	100
download_source()	183
construct a zipfile with the current source and stream it to the visitor	
download_source_tree()	184
workhorse function to recursively add most of a tree to a ZIP-archive	
DIALOG_NODE_EDIT	100
DIALOG_NODE_DELETE_CONFIRM	100
dialog_validate()	62
validate and check values that were submitted via a user dialog	
donors.php	65
/program/lib/donors/donors.php - a list of benefactors (people and organisations)	
DIALOG_NODE_ADD	100
DatabaseMysql	266
MySQL database	
DatabaseMysql::\$charset	266
DatabaseMysql::\$db_type	267
DatabaseMysql::\$db_username	268
DatabaseMysql::\$db_version	268
DatabaseMysql::\$db_server	267
DatabaseMysql::\$db_password	267

DatabaseMysql::\$collation	266
DatabaseMysql::\$db_link	267
DatabaseMysql::\$db_name	267
diff_microtime()	11

Calculate the difference between two microtimes

E

Email::rfc5322_address()	309
<i>construct an address field according to RFC5322 (RFC822)</i>	
Email::rfc5322_message_id()	310
<i>construct a message-id conforming to RFC5322 (RFC2822, RFC822)</i>	
Email::send()	311
<i>send the message using the prepared information (To:, Subject:, the message and attachments etc.)</i>	
Email::send_body()	311
<i>actually send a prepared mail body (with headers) to recipient(s)</i>	
Email::rfc2047_qstring()	307
<i>encode a string according to RFC2047 (Message Header Extensions for Non-ASCII Text)</i>	
Email::rfc2047_qchar()	306
<i>encode an 8-bit byte according to Q-encoding in RFC2047</i>	
Email::encode_part_headers()	304
<i>construct necessary headers for a MIME body part</i>	
Email::is_7bit()	305
<i>a small utility routine to determine if a string has only 7bit characters</i>	
Email::prepare_body()	305
<i>construct the full mail body and the necessary top-level mail header fields</i>	
Email::reset_all()	306
<i>reset all variables to their default values</i>	
Email::set_header()	312
<i>manually add a header to the mail message</i>	
Email::set_mailfrom()	312
<i>record the address and the name for the From: header</i>	
en_manifest.php	585
<i>/program/languages/en/en_manifest.php - description of the main language/translation (English)</i>	
es_manifest.php	590
<i>/program/languages/es/es_manifest.php - description of the Spanish translation</i>	
error_handler()	816
escape_quote()	816
el_manifest.php	570
<i>/program/languages/el/el_manifest.php - description of the Greek translation</i>	
Email::set_subject()	314
<i>store the subject of the mail message</i>	
Email::set_mailreplyto()	313
<i>record the address and the name for the Reply-To: header</i>	
Email::set_mailto()	313
<i>record the address and the name for the To: header</i>	
Email::set_message()	313
<i>set the (primary) message</i>	
Email::encode_part()	304
<i>encode an email body part with headers and all</i>	

Email::encode_message()	303
<i>encode the main message, optionally including 1 or more alternative versions</i>	
Email::\$headers	298
Email::\$mailcc	299
Email::\$mailfrom	299
Email::\$mailreplyto	299
Email::\$eol	298
Email::\$charset	298
email.class.php	66
<i>/program/lib/email.class.php - wrapper for sending mail</i>	
error_exit404()	184
<i>exit with a 404 not found error</i>	
Email	297
<i>Email implements a simple interface to send mail</i>	
Email::\$attachments	298
Email::\$mailto	299
Email::\$max_length	299
Email::add_mailcc()	301
<i>add an address and name for the Cc: header</i>	
Email::add_message()	301
<i>add an (alternative version of) message</i>	
Email::add_related()	302
<i>add a related attachment</i>	
Email::boundary()	303
<i>construct a unique boundary for use within this mail message</i>	
Email::add_attachment()	300
<i>add an attachment</i>	
Email::\$subject	300
Email::\$messages	300
Email::\$minimal	300
Email::\$related	300
error_exit()	11
<i>emergency exit of program in case there is something really, really wrong</i>	

F

frugal.php	613
<i>/program/themes/frugal/languages/fa/frugal.php</i>	
fi_manifest.php	620
<i>/program/languages/fi/fi_manifest.php - description of the Finnish translation</i>	
frugal.php	626
<i>/program/themes/frugal/languages/fi/frugal.php</i>	
fa_manifest.php	607
<i>/program/languages/fa/fa_manifest.php - description of the Persian translation</i>	
frugal.php	599
<i>/program/themes/frugal/languages/es/frugal.php</i>	
frugal.php	562
<i>/program/themes/frugal/languages/de/frugal.php</i>	
frugal.php	581
<i>/program/themes/frugal/languages/el/frugal.php</i>	
fil_manifest.php	631
<i>/program/languages/fil/fil_manifest.php - description of the Filipino translation</i>	

fr_manifest.php	638
<i>/program/languages/fr/fr_manifest.php - description of the French translation</i>	
frugal.php	713
<i>/program/themes/frugal/languages/it/frugal.php</i>	
frugal.php	735
<i>/program/themes/frugal/languages/pap/frugal.php</i>	
frugal.php	694
<i>/program/themes/frugal/languages/hu/frugal.php</i>	
frugal.php	672
<i>/program/themes/frugal/languages/fry/frugal.php</i>	
frugal.php	650
<i>/program/themes/frugal/languages/fr/frugal.php</i>	
fry_manifest.php	658
<i>/program/languages/fry/fry_manifest.php - description of the Western Frisian translation</i>	
frugal.php	543
<i>/program/themes/frugal/languages/da/frugal.php</i>	
frugal.php	529
<i>/program/themes/frugal/languages/bg/frugal.php</i>	
FileManager::task_preview_file()	334
<i>preview a file via file.php</i>	
FileManager::task_remove_directory()	334
<i>show a confirmation dialog for removing a single directory OR actually removes a directory</i>	
FileManager::task_list_directory()	333
<i>show a directory listing of the current working directory and links to add/delete files/directories etc.</i>	
FileManager::task_change_directory()	333
<i>make another directory the current (working) directory and optionally change the sort order</i>	
FileManager::task_add_directory()	333
<i>create a new subdirectory</i>	
FileManager::task_add_file()	333
<i>add one or more new files to a directory</i>	
FileManager::task_remove_file()	334
<i>show a confirmation dialog for deleting a single file</i>	
FileManager::task_remove_multiple_files()	334
<i>show confirmation dialog for multiple file delete OR perform actual file delete</i>	
FileManager::vpath()	337
<i>translate a path to the corresponding virtual path</i>	
frugal.php	509
<i>/program/themes/frugal/languages/ar/frugal.php</i>	
FileManager::vname()	337
<i>construct the (possibly translated) name of the last directory in the path</i>	
FileManager::virusscan()	336
<i>scan a file for viruses</i>	
FileManager::unique_filename()	335
<i>construct a unique filename taking existing files into account</i>	
FileManager::valid_path()	335
<i>access control and validation for selected directory or file</i>	
frugal.php	749
<i>/program/themes/frugal/languages/pl/frugal.php</i>	
frugal.php	768
<i>/program/themes/frugal/languages/pt/frugal.php</i>	
frugal.php	905
<i>/program/themes/frugal/languages/vi/frugal.php</i>	

frugal.php	919
<i>/program/themes/frugal/languages/zh/frugal.php</i>	
frugal_install.php	1268
<i>/program/themes/frugal/frugal_install.php -- installer of the frugal theme</i>	
frugal.php	883
<i>/program/themes/frugal/languages/ur/frugal.php</i>	
frugal.php	855
<i>/program/themes/frugal/languages/sv/frugal.php</i>	
FCKEditor::GetConfigFieldString()	838
<i>Get settings from Config array as a single string.</i>	
FCKEditor::IsCompatible()	838
<i>Returns true if browser is compatible with FCKEditor.</i>	
frugal_demodata()	1268
<i>add demonstration data to the system</i>	
frugal_install()	1269
<i>install the theme</i>	
frugal.php	1273
<i>/program/themes/frugal/languages/nl/frugal.php - translated messages for theme (Nederlands)</i>	
FAQ	1337
frugal.php	1272
<i>/program/themes/frugal/languages/en/frugal.php - translated messages for theme (English)</i>	
frugal_manifest.php	1271
<i>/program/themes/frugal/frugal_manifest.php - description of the frugal theme</i>	
frugal_uninstall()	1269
<i>uninstall the theme</i>	
frugal_upgrade()	1270
<i>upgrade the theme</i>	
FCKEditor::EncodeConfig()	837
<i>Encode characters that may break the configuration string generated by GetConfigFieldString().</i>	
FCKEditor::CreateHtml()	837
<i>Return the HTML code required to run FCKEditor.</i>	
FCKEditor_IsCompatibleBrowser()	819
<i>Check if browser is compatible with FCKEditor.</i>	
<i>Return true if is compatible.</i>	
FCKEditor	836
fckeditor_php4.php	819
fckeditor.php	818
frugal.php	780
<i>/program/themes/frugal/languages/ro/frugal.php</i>	
frugal.php	798
<i>/program/themes/frugal/languages/ru/frugal.php</i>	
FCKEditor::\$BasePath	836
<i>Path to FCKEditor relative to the document root.</i>	
FCKEditor::\$Config	836
<i>This is where additional configuration can be passed.</i>	
FCKEditor::\$Width	837
<i>Width of the FCKEditor.</i>	
<i>Examples: 100%, 600</i>	
FCKEditor::Create()	837
<i>Display FCKEditor.</i>	
FCKEditor::\$Value	837
<i>Initial value.</i>	

FCKeditor::\$ToolbarSet	837
<i>Name of the toolbar to load.</i>	
FCKeditor::\$Height	836
<i>Height of the FCKeditor.</i>	
<i>Examples: 400, 50%</i>	
FCKeditor::\$InstanceName	836
<i>Name of the FCKeditor instance.</i>	
FileManager::sort_entries()	332
<i>sort directory entries</i>	
FileManager::show_menu()	332
<i>show a menu that is equivalent with the root directory</i>	
FileManager::\$current_directory	315
FileManager::\$ext_allow_browse	315
FileManager::\$ext_allow_upload	315
FileManager::\$areas	315
FileManager	314
<i>File Manager</i>	
filemanager.class.php	71
<i>/program/lib/filemanager.class.php - filemanager</i>	
friendly_bookmark()	148
<i>construct an alphanumeric string from a (node) title yielding a readable bookmark filename</i>	
FileManager::\$job	316
FileManager::\$output	316
FileManager::allowed_extensions()	317
<i>convert a comma-delimited list of allowable extensions to an array (or FALSE if none are allowed)</i>	
FileManager::cmp_entries_bydate_asc()	318
<i>callback for comparing two directory entries by mtime</i>	
FileManager::\$vpaths	317
FileManager::\$usergroups	316
FileManager::\$show_thumbnails	316
FileManager::\$sort	316
file_available()	67
<i>check availability of requested file for the current user</i>	
filelib.php	67
<i>/program/lib/filelib.php - utilities for manipulating files</i>	
F_DATETIME	53
F_FILE	53
F_DATE	53
F_CSRFTOKEN	53
F_ALPHANUMERIC	52
F_CHECKBOX	53
F_INTEGER	53
F_LISTBOX	53
F_SUBMIT	53
F_TIME	53
F_RICHTEXT	53
F_REAL	53
F_PASSWORD	53
F_RADIO	53
FileManager::cmp_entries_bydate_desc()	318
<i>callback for comparing two directory entries by date (descending)</i>	
FileManager::cmp_entries_byfile_asc()	318

<i>callback for comparing two directory entries by filename</i>	325
FileManager::sanitise filetype()	325
<i>try to make sure that the extension of file \$name makes sense or matches the actual filetype</i>	
FileManager::save_uploaded_file()	326
<i>move the uploaded file in place or perhaps resize it first (supported images only) + create thumb</i>	
FileManager::human_readable_size()	325
<i>convert an integer filesize to a human readable form</i>	
FileManager::has_allowed_extension()	325
<i>see if the filename extension is allowed</i>	
FileManager::get_entries_users()	324
<i>generate a list of (virtual) directories for users this user can access</i>	
FileManager::get_max_file_uploads()	325
<i>maximum number of files accepted in a single upload (since PHP 5.2.12)</i>	
FileManager::show_breadcrumbs()	328
<i>display a clickable path to the directory \$path</i>	
FileManager::show_dialog_add_files()	328
<i>display the file upload dialog</i>	
FileManager::show_file_as_thumbnail()	331
<i>show a thumbnail of a single (image) file perhaps including clickable links for use in FCK Editor</i>	
FileManager::show_list()	332
<i>display a list of directories and files in \$path</i>	
FileManager::show_directories_and_files()	330
<i>display a list of subdirectories and files in directory \$path</i>	
FileManager::show_directories()	329
<i>output a simple list of directories (for navigation only)</i>	
FileManager::show_dialog_confirm_delete_directory()	328
<i>show a dialog that ask the user to confirm the removal of a directory</i>	
FileManager::show_dialog_confirm_delete_files()	329
<i>show a dialog that ask the user to confirm a mass file delete</i>	
FileManager::get_entries_root()	324
<i>generate a list of (virtual) directories at the root level</i>	
FileManager::get_entries_groups()	324
<i>generate a list of (virtual) directories for groups the user can access</i>	
FileManager::delete_directory()	320
<i>workhorse function that actually removes directories</i>	
FileManager::delete_files()	321
<i>workhorse function that actually deletes files, and possibly the corresponding thumbnails</i>	
FileManager::cmp_groups()	320
<i>callback for comparing two group records</i>	
FileManager::cmp_entries_bysize_desc()	319
<i>callback for comparing two directory entries by size (descending)</i>	
FileManager::cmp_entries_byfile_desc()	319
<i>callback for comparing two directory entries by filename (descending)</i>	
FileManager::cmp_entries_bysize_asc()	319
<i>callback for comparing two directory entries by size</i>	
FileManager::explode_path()	321
<i>shorthand for splitting a path into an array with path components</i>	
FileManager::file_url()	321
<i>construct a url that links to a file via /file.php</i>	
FileManager::get_entries()	323
<i>generate a list of selected files and subdirectories in \$path</i>	
FileManager::get_entries_areas()	323

generate a list of (virtual) directories for areas the user can access	
FileManager::get_dialogdef_confirm_delete_files()	323
construct a dialog definition for removing/deleting files	
FileManager::get_dialogdef_confirm_delete_directory()	322
construct a dialog definition for removing/deleting a subdirectory	
FileManager::get_dialogdef_add_directory()	322
construct a dialog definition for adding a subdirectory	
FileManager::get_dialogdef_add_files()	322
construct a dialog definition for adding (uploading) files	
file.php	8
/file.php - the main entrypoint for serving files	

G

GroupManager::get_group_record()	343
retrieve a single group's record possibly from the cache	
GroupManager::get_group_capacity_records()	343
return an array of group-capacity records (possibly buffered)	
GroupManager::get_group_capacity_names()	342
shortcut to retrieve the name and full name of the selected group and optionally a capacity name	
GroupManager::get_icon_delete()	343
construct a clickable icon to delete this group	
GroupManager::get_icon_edit()	344
construct a clickable icon to edit the properties of this group	
GroupManager::group_add()	345
present 'add group' dialog where the user can enter minimal properties for a new group	
GroupManager::groups_overview()	344
display list of existing groups and an option to add a group	
GroupManager::get_options_capacities()	344
construct a simple option list with all available capacity names keyed by capacity code	
GroupManager::get_groupname()	342
shorthand to get the name of a group	
GroupManager::get_dialogdef_edit_group()	342
construct the edit group dialog	
GroupManager::capacity_intranet()	340
show a dialog for modifying intranet permissions for a group/capacity	
GroupManager::capacity_admin()	340
show a dialog for modifying admin permissions for a group/capacity	
GroupManager::calc_acl_id()	340
retrieve the acl_id for a particular group/capacity from the database	
GroupManager::capacity_overview()	341
display an overview of all members of a group with a particular capacity	
GroupManager::capacity_pagemanager()	341
show a dialog for modifying page manager permissions for a group/capacity	
GroupManager::get_dialogdef_add_group()	342
construct the add group dialog	
GroupManager::delete_group_capacities_records()	341
actually remove a group and all associated records	
GroupManager::capacity_save()	341
save data from a dialog for a group/capacity	
GroupManager::group_delete()	345

<i>delete a group after confirmation</i>	
GroupManager::group_edit()	346
<i>show a dialog with the basic properties of a group</i>	
guestbook_connect()	1033
<i>connect this module to a node</i>	
guestbook_admin.php	1033
<i>/program/modules/guestbook/guestbook_admin.php - management interface for module</i>	
get_org_property()	820
guestbook_disconnect()	1034
<i>disconnect this module from a node</i>	
guestbook_save()	1034
<i>save the modified content data of this module linked to node \$node_id</i>	
guestbook.php	1038
<i>/program/modules/guestbook/languages/nl/guestbook.php - translated messages for module (Dutch)</i>	
guestbook.php	1037
<i>/program/modules/guestbook/languages/en/guestbook.php - translated messages for module (English)</i>	
guestbook_show_edit()	1035
<i>present the user with a dialog to modify the content that is connected to node \$node_id</i>	
GroupManager::valid_group_capacity()	349
<i>shorthand to test the validity of a particular group/capacity</i>	
GroupManager::show_parent_menu()	349
GroupManager::has_job_permission()	347
<i>determine whether a group/capacity has permissions for a particular job</i>	
GroupManager::group_savenew()	346
<i>save a new group to the database</i>	
GroupManager::group_save()	346
<i>save an edited group to the database, including adding/modifying/deleting group/capacity-records</i>	
GroupManager::show_breadcrumbs_addgroup()	347
<i>display breadcrumb trail that leads to the add new group dialog</i>	
GroupManager::show_breadcrumbs_group()	347
<i>display breadcrumb trail that leads to groups overview screen</i>	
GroupManager::show_menu_groupcapacity()	348
<i>show a menu for a group capacity with options to modify privileges, etc.</i>	
GroupManager::show_menu_group()	348
<i>show a menu for a group including links to the group's capacity overview screens</i>	
GroupManager::show_breadcrumbs_groupcapacity()	348
<i>display breadcrumb trail that leads to group capacity overview screen</i>	
GroupManager::a_params()	339
<i>shorthand for the anchor parameters that lead to the group manager</i>	
GroupManager::areas_expand_collapse()	339
<i>manipulate the current state if indicator(s) for 'open' and 'closed' areas</i>	
get_area_records()	148
<i>retrieve a list of all available area records keyed by area_id</i>	
GROUP_SELECT_NO_GROUP	138
<i>this value is used to select the users that are not associated with any group</i>	
GROUP_SELECT_ALL_USERS	138
<i>this value is used to select all users rather than users from a specific group</i>	
get_cookie_string()	149
<i>return an (unquoted) string value specified in the cookie header or default value if none</i>	
get_csrf_token()	149

get csrf token name and value	150
get_module_records()	150
<i>retrieve a list of all available module records</i>	
get_friendly_parameter()	150
<i>retrieve a named parameter from the friendly URL</i>	
get_editor_names()	149
<i>prepare a list of available editors</i>	
get_manifests()	122
<i>retrieve an array of manifests for modules, themes or languages</i>	
GROUPMANAGER_MAX_CAPACITIES	72
<i>this defines the maximum number of capacities a group can have (keep this below 10 because of dialog hotkeys)</i>	
GROUPMANAGER_DIALOG_CAPACITY_PAGEMANAGER	21
GROUPMANAGER_DIALOG_CAPACITY_INTRANET	21
GROUPMANAGER_DIALOG_CAPACITY_ADMIN	21
GROUPMANAGER_DIALOG_DELETE	21
GROUPMANAGER_DIALOG_EDIT	21
groupmanager.class.php	72
<i>/program/lib/groupmanager.class.php - taking care of group management</i>	
get_mimetype()	68
<i>determine the mimetype of a file</i>	
get_mediatype()	67
<i>extract the mediatype and -subtype from a full mimetype</i>	
get_page_address_url()	151
<i>return the reconstructed URL in a single (indented) line</i>	
get_parameter_int()	151
<i>return an integer value specified in the page request or default value if none</i>	
GroupManager	337
<i>Group management</i>	
get_available_manuallys()	194
<i>construct a list of 0 or more languages of available manuals</i>	
get_available_languages()	194
<i>construct a list of 0 or more languages from the languages directory</i>	
GroupManager::\$groups	338
GroupManager::\$group_capacity_records	338
GroupManager::add_group_capacity()	339
<i>add a group/capacity and corresponding acl to the database</i>	
GroupManager::\$show_parent_menu	338
GroupManager::\$soutput	338
get_versioncheck_url()	179
<i>construct URL for version check against the project's website</i>	
get_current_skin()	178
<i>determine the default skin to use</i>	
get_requested_area()	153
<i>get the number of the area the user requested or null if not specified</i>	
get_properties()	152
<i>retrieve typed properties (name-value-pairs) from a table</i>	
get_parameter_string()	152
<i>return an (unquoted) string value specified in the page request or default value if none</i>	
get_requested_filename()	153
<i>get the name of the requested file</i>	
get_requested_node()	153
<i>get the number of the node the user requested or NULL if not specified</i>	

get_user_groups()	154
<i>retrieve the records of the groups of which user \$user_id is a member</i>	
get_unique_number()	154
<i>a small utility routine that returns a unique integer</i>	
get_skin_names()	154
<i>prepare a list of available skins</i>	
GROUPMANAGER_DIALOG_ADD	21
<i>Distinguish between the various dialogs</i>	

H

htmlpage.php	916
<i>/program/modules/htmlpage/languages/zh/htmlpage.php</i>	
htmlpage.php	898
<i>/program/modules/htmlpage/languages/vi/htmlpage.php</i>	
htmlpage.php	876
<i>/program/modules/htmlpage/languages/ur/htmlpage.php</i>	
htmlpage_admin.php	1040
<i>/program/modules/htmlpage/htmlpage_admin.php - management interface for htmlpage-module</i>	
htmlpage_connect()	1040
<i>connect this module to a node</i>	
htmlpage_get_dialogdef()	1041
<i>construct the dialog definition for editing a plain HTML page</i>	
htmlpage_disconnect()	1041
<i>disconnect this module from a node</i>	
htmlpage.php	848
<i>/program/modules/htmlpage/languages/sv/htmlpage.php</i>	
hmac()	820
htmlpage.php	761
<i>/program/modules/htmlpage/languages/pt/htmlpage.php</i>	
htmlpage.php	746
<i>/program/modules/htmlpage/languages/pl/htmlpage.php</i>	
htmlpage.php	728
<i>/program/modules/htmlpage/languages/pap/htmlpage.php</i>	
htmlpage.php	777
<i>/program/modules/htmlpage/languages/ro/htmlpage.php</i>	
htmlpage.php	791
<i>/program/modules/htmlpage/languages/ru/htmlpage.php</i>	
hexdump()	820
htmlpage.php	813
<i>/program/modules/htmlpage/languages/sk/htmlpage.php</i>	
htmlpage_save()	1041
<i>save the modified content data of this module linked to node \$node_id</i>	
htmlpage_show_edit()	1042
<i>present the user with a dialog to modify the content that is connected to node \$node_id</i>	
htmlpage_view()	1052
<i>display the content of the htmlpage linked to node \$node_id</i>	
htmlpage_view.php	1052
<i>/program/modules/htmlpage/htmlpage_view.php - interface to the view-part of the htmlpage module</i>	
htmlpage_search()	1049

search all <code>htmlpage</code> s linked to selected nodes for keywords in <code>\$qwords</code>	1053
htmlpage_tabledefs.php	1053
<i>/program/modules/htmlpage/install/htmlpage_tabledefs.php - data definition for module</i>	
htmlpage.php	1054
<i>/program/modules/htmlpage/languages/en/htmlpage.php - translated messages for module (English)</i>	
HISTORY	1342
htmlpage.php	1055
<i>/program/modules/htmlpage/languages/nl/htmlpage.php - translated messages for module (Dutch)</i>	
htmlpage_search.php	1049
<i>/program/modules/htmlpage/htmlpage_search.php - interface to the search-part of this module</i>	
htmlpage_manifest.php	1048
<i>/program/modules/htmlpage/htmlpage_manifest.php - description of the htmlpage module</i>	
htmlpage_install.php	1045
<i>/program/modules/htmlpage/htmlpage_install.php - installer of the htmlpage module</i>	
htmlpage_cron()	1044
<i>routine that is called periodically by cron</i>	
htmlpage_cron.php	1044
<i>/program/modules/htmlpage/htmlpage_cron.php - interface to the cron-part of the htmlpage module</i>	
htmlpage_demodata()	1045
<i>add demonstration data to the system</i>	
htmlpage_install()	1046
<i>install the module</i>	
htmlpage_upgrade()	1046
<i>upgrade the module</i>	
htmlpage_uninstall()	1046
<i>uninstall the module</i>	
htmlpage.php	706
<i>/program/modules/htmlpage/languages/it/htmlpage.php</i>	
htmlpage.php	691
<i>/program/modules/htmlpage/languages/hu/htmlpage.php</i>	
html_table_cell()	77
html_table()	77
<i>construct the opening of a HTML table</i>	
html_input_text()	76
<i>STUB</i>	
html_table_cell_close()	77
html_table_close()	77
<i>construct table closing tag</i>	
html_table_head_close()	77
html_table_head()	77
html_input_submit()	76
<i>STUB</i>	
html_input_select()	76
<i>STUB</i>	
html_attributes()	74
<i>convert an array of name-value pairs to a string</i>	
html_a()	73
<i>construct an HTML A tag with optional parameters and attributes and optional fragment</i>	
href()	73
<i>construct a href from a path, params and a fragment</i>	

html_form()	75
construct the opening of a HTML form	
html_form_close()	75
companion of <code>html_form</code> : close the tag	
html_input_radio()	76
STUB	
html_img()	75
construct an HTML IMG tag with optional attributes	
html_table_row()	77
html_table_row_close()	78
htmlpage.php	643
/program/modules/htmlpage/languages/fr/htmlpage.php	
htmlpage.php	623
/program/modules/htmlpage/languages/fi/htmlpage.php	
htmlpage.php	610
/program/modules/htmlpage/languages/fa/htmlpage.php	
htmlpage.php	665
/program/modules/htmlpage/languages/fry/htmlpage.php	
hi_manifest.php	679
/program/languages/hi/hi_manifest.php - description of the Hindi translation	
hu_manifest.php	688
/program/languages/hu/hu_manifest.php - description of the Hungarian translation	
htmlpage.php	682
/program/modules/htmlpage/languages/hi/htmlpage.php	
htmlpage.php	595
/program/modules/htmlpage/languages/es/htmlpage.php	
htmlpage.php	577
/program/modules/htmlpage/languages/el/htmlpage.php	
hmac()	155
calculate hmac according to RFC2104 (February 1997)	
html_tag_close()	78
companion of <code>html_tag</code> : close the tag	
html_tag()	78
construct a generic HTML-tag with attributes, optionally close it too	
htmlpage.php	506
/program/modules/htmlpage/languages/ar/htmlpage.php	
htmlpage.php	522
/program/modules/htmlpage/languages/bg/htmlpage.php	
htmlpage.php	555
/program/modules/htmlpage/languages/de/htmlpage.php	
htmlpage.php	540
/program/modules/htmlpage/languages/da/htmlpage.php	
htmlilib.php	73
/program/lib/htmlilib.php - useful functions for generating HTML-code	

InstallWizard::save_installtype()	491
store the selected install type + high visibility flag	
InstallWizard::save_language()	491
store the selected language	
InstallWizard::save_database()	490

<i>validate database information</i>	490
<u>InstallWizard::save_cms()</u>	490
<i>validate and store the CMS-data the user supplied</i>	
<u>InstallWizard::run()</u>	489
<i>main dispatcher for the Installation Wizard</i>	
<u>InstallWizard::sanitise_filename()</u>	489
<i>sanitise a string to make it acceptable as a filename/directoryname</i>	
<u>InstallWizard::save_user()</u>	491
<i>validate and store the data for the first user account</i>	
<u>InstallWizard::show_dialog_cancelled()</u>	491
<i>show the user that the process has been cancelled</i>	
<u>InstallWizard::show_dialog_finish()</u>	494
<i>construct the finish screen</i>	
<u>InstallWizard::show_dialog_installtype()</u>	494
<i>construct the installtype + high visibility selection dialog</i>	
<u>InstallWizard::show_dialog_database()</u>	493
<i>construct the dialog for database (server, host, username, password, etc.)</i>	
<u>InstallWizard::show_dialog_confirm()</u>	493
<i>construct the overview/confirmation dialog</i>	
<u>InstallWizard::show_dialog_cms()</u>	492
<i>construct the dialog for essential cms data (title, paths, e-mail address)</i>	
<u>InstallWizard::show_dialog_compatibility()</u>	492
<i>construct the comptibility overview</i>	
<u>InstallWizard::render_dialog()</u>	488
<i>quick and dirty dialogdef renderer</i>	
<u>InstallWizard::quasi_random_string()</u>	488
<i>generate a string with quasi-random characters</i>	
<u>InstallWizard::mysql_get_server_info()</u>	484
<i>unified mysql/mysqli wrapper for get_server_info</i>	
<u>InstallWizard::mysql_num_rows()</u>	485
<i>unified mysql/mysqli wrapper for num_rows</i>	
<u>InstallWizard::mysql_connect()</u>	484
<i>unified mysql/mysqli wrapper for connect</i>	
<u>InstallWizard::mysql_close()</u>	484
<i>unified mysql/mysqli wrapper for close</i>	
<u>InstallWizard::magic_unquote()</u>	484
<i>this circumvents the 'magic' in magic_quotes_gpc() by conditionally stripping slashes</i>	
<u>InstallWizard::microtime()</u>	484
<i>a trick for systems without the microtime function</i>	
<u>InstallWizard::mysql_query()</u>	485
<i>unified mysql/mysqli wrapper for query</i>	
<u>InstallWizard::mysql_real_escape_string()</u>	485
<i>unified mysql/mysqli wrapper for real_escape_string</i>	
<u>InstallWizard::mysql_utf8_support()</u>	486
<i>determine the level of UTF-8 support based on MySQL-server version</i>	
<u>InstallWizard::perform_installation()</u>	487
<i>perform the actual initialisation of the cms</i>	
<u>InstallWizard::mysql_utf8mb3()</u>	486
<i>massage string to contain only 3-byte UTF8-sequences</i>	
<u>InstallWizard::mysql_set_charset()</u>	486
<i>unified mysql/mysqli wrapper for set_charset</i>	
<u>InstallWizard::mysql_select_db()</u>	486
<i>unified mysql/mysqli wrapper for select_db</i>	

InstallWizard::show_dialog_language()	494
construct the language selection dialog	
InstallWizard::show_dialog_license()	495
construct a full license agreement and an input where the user must enter 'I agree'	
install.php	741
/program/install/languages/pl/install.php	
install.php	754
/program/install/languages/pt/install.php	
install.php	721
/program/install/languages/pap/install.php	
it_manifest.php	701
/program/languages/it/it_manifest.php - description of the Italian translation	
install.php	686
/program/install/languages/hu/install.php	
install.php	699
/program/install/languages/it/install.php	
install.php	784
/program/install/languages/ru/install.php	
initialise()	821
install.php	911
/program/install/languages/zh/install.php	
INSTALL	1342
install.php	889
/program/install/languages/vi/install.php	
install.php	869
/program/install/languages/ur/install.php	
install.php	841
/program/install/languages/sv/install.php	
install.php	860
/program/install/languages/tr/install.php	
install.php	656
/program/install/languages/fry/install.php	
install.php	636
/program/install/languages/fr/install.php	
InstallWizard::write_config_php()	498
attempt to write the file config.php in the correct location	
install.php	501
/program/install/languages/ar/install.php	
InstallWizard::validate_password()	497
validation of password input	
InstallWizard::validate()	496
minimal validation of data input	
InstallWizard::show_dialog_user()	495
construct the dialog for the first user account	
InstallWizard::t()	496
retrieve a translated string with optional parameters filled in	
install.php	514
/program/install/languages/bg/install.php	
install.php	535
/program/install/languages/da/install.php	
install.php	605
/program/install/languages/fa/install.php	
install.php	618

/program/install/languages/fi/install.php	588
install.php	588
/program/install/languages/es/install.php	568
install.php	568
/program/install/languages/el/install.php	548
install.php	548
/program/install/languages/de/install.php	483
InstallWizard::is_already_installed()	483
<i>check for previous install</i>	
InstallWizard::invisible_test_image()	482
<i>create a link to an invisible image to test the friendly URL feature</i>	
INSTALL_DIALOG_LICENSE	455
INSTALL_DIALOG_USER	456
INSTALL_DIALOG_LANGUAGE	455
INSTALL_DIALOG_INSTALLTYPE	455
INSTALL_DIALOG_DOWNLOAD	455
INSTALL_DIALOG_FINISH	455
install_script_name()	456
<i>determine the name of the executing script (the entry point)</i>	
index.php	462
<i>/program/install/index.php - redirector for website installation</i>	
InstallWizard::\$messages	471
InstallWizard::\$results	471
InstallWizard::\$license	471
InstallWizard	470
<i>class for performing installation tasks</i>	
install.php	464
<i>/program/install/languages/en/install.php - translated messages for /program/install.php (English)</i>	
install.php	466
<i>/program/install/languages/nl/install.php - translated messages for /program/install.php (Dutch)</i>	
INSTALL_DIALOG_DONE	455
INSTALL_DIALOG_DATABASE	455
install_theme()	123
<i>install an additional theme</i>	
ini_get_int()	155
<i>return an integer (bytecount) value from PHP ini</i>	
install_module()	123
<i>install an additional module</i>	
install_language()	122
<i>install an additional language pack</i>	
init.php	10
<i>/program/init.php - setup database connection, sessions, configuration, etc.</i>	
initialise()	13
<i>initialise the program, setup database, read configuration, etc.</i>	
is_expired()	155
<i>determine if any of the ancestors or \$node_id itself is already expired</i>	
is_under_embargo()	156
<i>determine if any of the ancestors or \$node_id itself is under embargo</i>	
INSTALL_DIALOG_COMPATIBILITY	455
INSTALL_DIALOG_CONFIRM	455
INSTALL_DIALOG_CMS	455
INSTALL_DIALOG_CANCELLED	455

install.php	455
<i>/program/install.php - the main entrypoint for website installation</i>	
InstallWizard::\$time_start	471
InstallWizard::appropriate_legal_notices()	472
<i>construct a link to appropriate legal notices as per AGPLv3 section 5</i>	
InstallWizard::get_dialogdef_language()	478
<i>fill an array with necessary information for language dialog</i>	
InstallWizard::get_dialogdef_user()	478
<i>fill an array with necessary information for the first user dialog</i>	
InstallWizard::get_dialogdef_installtype()	478
<i>fill an array with necessary information for installtype dialog</i>	
InstallWizard::get_dialogdef_finish()	477
<i>fill an array with necessary information for finish / jump dialog</i>	
InstallWizard::get_dialogdef_cms()	477
<i>fill an array with necessary information for the cms dialog</i>	
InstallWizard::get_dialogdef_database()	477
<i>fill an array with necessary information for the database dialog</i>	
InstallWizard::get_list_of_install_languages()	478
<i>retrieve a list of available languages by querying the file system for install.php translation files</i>	
InstallWizard::get_manifests()	478
<i>retrieve an array of manifests for modules, themes or languages</i>	
InstallWizard::guess_url()	482
<i>educated guesses for scheme, host and portname from \$_SERVER</i>	
InstallWizard::insert_tabledata()	482
<i>fill tables in database via include()'ing a file with tabledata</i>	
InstallWizard::get_utf8_parameter_string()	481
<i>return a valid (unquoted) UTF-8 string parameter typically from \$_POST, or default value if none</i>	
InstallWizard::get_page()	480
<i>construct a complete HTML-page that can be sent to the user's browser</i>	
InstallWizard::get_menu()	479
<i>construct a clickable menu which helps the user to jump back and forth in the funnel</i>	
InstallWizard::get_options_db_type()	480
<i>construct a list of database options</i>	
InstallWizard::get_default_install_values()	477
<i>return an array with default configuration values</i>	
InstallWizard::gd_supported()	476
<i>retrieve information about GD and supported graphics file formats</i>	
InstallWizard::check_validation()	474
<i>shorthand to check the validation status of the relevant dialogs</i>	
InstallWizard::clamscan_installed()	474
<i>try to locate clamdscan or clamscan on the server</i>	
InstallWizard::check_license()	473
<i>check if the user accepts the licences</i>	
InstallWizard::check_for_nameclash()	473
<i>check for name clash of new user (webmaster) and user accounts from demodata</i>	
InstallWizard::button()	473
<i>shorthand for creating a submit button in the correct style</i>	
InstallWizard::check_compatibility()	473
<i>check certain compatibility issues and optionally return test results</i>	
InstallWizard::construct_config_php()	474
<i>prepare a configuration file based on the collected information</i>	
InstallWizard::create_tables()	474

<i>create tables in database via include()'ing a file with tabledefs</i>	475
InstallWizard::errorcount_reset()	475
<i>reset the error counter</i>	
InstallWizard::fetch_license()	475
<i>helper to retrieve the text of the LICENSE AGREEMENT for Website</i>	
InstallWizard::errorcount_bump()	475
<i>increment the error counter and perhaps slow things down</i>	
InstallWizard::end_session_and_redirect()	475
<i>unset installation data, end session and redirect the user to elsewhere</i>	
InstallWizard::diff_microtime()	475
<i>Calculate the difference between two microtimes (borrowed from init.php)</i>	
index.php	9
<i>/index.php - the main entryptpoint for website visitors (frontpage)</i>	

J

JOB_CONFIGURATIONMANAGER	175
<i>This is used to dispatch the configuration manager</i>	
JOB_FILEBROWSER	175
<i>This is used to dispatch the file manager in file browser mode (used with CKEditor and FCKeditor)</i>	
JOB_FILEMANAGER	175
<i>This is used to dispatch the file manager</i>	
JOB_ACCOUNTMANAGER	175
<i>This is used to dispatch the account manager (users and groups)</i>	
javascript_alert()	157
<i>message a message and generate a javascript alert()</i>	
JOB_PERMISSION_TOOLS	137
<i>combine the permssions for the tools in a single bit mask for convenient testing</i>	
JOB_PERMISSION_TRANSLATETOOL	137
<i>This allows the user to translate the program, by modifying existing translations or adding new languages</i>	
JOB_PERMISSION_UPDATE	137
<i>This allows the user to perform a system upgrade (see also was_version_check() and main_admin())</i>	
JOB_FLASHBROWSER	175
<i>This is used to dispatch the file manager in flash browser mode (used with CKEditor and FCKeditor)</i>	
JOB_IMAGEBROWSER	175
<i>This is used to dispatch the file manager in image browser mode (used with CKEditor and FCKeditor)</i>	
JOB_TOOLS	176
<i>This is used to dispatch the tool manager</i>	
JOB_UPDATE	176
<i>This is used to dispatch the update manager</i>	
job_start()	179
<i>generate the start centre page</i>	
JOB_STATISTICS	176
<i>This is used to dispatch the statistics</i>	
JOB_STARTCENTER	176
<i>This is used to dispatch the startcenter job</i>	
JOB_MODULEMANAGER	176

<i>This is used to dispatch the module manager</i>	
JOB_PAGEMANAGER	176
<i>This is used to dispatch the page manager</i>	
JOB_PERMISSION_STATISTICS	136
<i>This permissions allows the user to access the site statistics</i>	
JOB_PERMISSION_STARTCENTER	136
<i>This permission is required for every user that is to logon to admin.php</i>	
job_update()	124
<i>main entry point for update wizard (called from /program/main_admin.php)</i>	
JOB_PERMISSION_ACCOUNTMANAGER	135
<i>This (dangerous) permission allows access to add/edit/delete users and groups (including escalate privileges)</i>	
JOB_PERMISSION_BACKUPTOOL	135
<i>This allows the user to download a backup of the database</i>	
job_tools()	113
<i>main entry point for tools (called from /program/main_admin.php)</i>	
job_statistics()	102
<i>main entry point for statistics (called from admin.php)</i>	
job_configurationmanager()	29
<i>main entry point for configurationmanager (called from /program/main_admin.php)</i>	
job_modulemanager()	97
<i>main entry point for modulemanager (called from /program/main_admin.php)</i>	
JOB_PERMISSION_CONFIGURATIONMANAGER	136
<i>This permission allows the user to access the configuration manager and change the site configuration</i>	
JOB_PERMISSION_FILEMANAGER	136
<i>This permission allows the user to access the file manager and upload/delete files in selected places</i>	
JOB_PERMISSION_NEXT_AVAILABLE_VALUE	136
<i>NOTE: This quasi-permission should always be defined to be the highest permission < 1</i>	
JOB_PERMISSION_PAGEMANAGER	136
<i>This permission allows the user to access the page manager and add/edit/delete nodes according to the user's ACLs</i>	
JOB_PERMISSION_SESSIONTOOL	136
<i>This allows the user to forcefully remove sessions [2016-05-18]</i>	
JOB_PERMISSION_MODULEMANAGER	136
<i>This permission allows the user to access the module manager and configure modules</i>	
JOB_PERMISSION_MASK	136
<i>This mask can be used to isolate only the 'official' permissions from an integer value</i>	
JOB_PERMISSION_GURU	136
<i>Guru permissions = all permission bits are set, even the unused ones</i>	
JOB_PERMISSION_LOGVIEW	136
<i>This allows the user to view the contents of the log table</i>	
job_accountmanager()	22
<i>main entry point for accountmanager (called from admin.php)</i>	

L

loginlib.php	571
<i>/program/languages/el/loginlib.php</i>	
loginlib.php	551
<i>/program/languages/de/loginlib.php</i>	

loginlib.php	538
/program/languages/da/loginlib.php	
loginlib.php	591
/program/languages/es/loginlib.php	
loginlib.php	608
/program/languages/fa/loginlib.php	
loginlib.php	639
/program/languages/fr/loginlib.php	
loginlib.php	632
/program/languages/fil/loginlib.php	
loginlib.php	621
/program/languages/fi/loginlib.php	
loginlib.php	517
/program/languages/bg/loginlib.php	
loginlib.php	504
/program/languages/ar/loginlib.php	
Language::get_phrase()	353
translation of phrases via a phrase key	
Language::get_languages_to_try()	352
calculate a list of possible languages and parent-languages to try for translations	
Language::get_filenames_to_try()	351
calculate an ordered list of filenames to try for translation of phrases	
Language::get_phrases_from_database()	354
retrieve phrases from the database for specified domain and language	
Language::get_phrases_from_file()	354
return the \$string array after including a file	
Language::retrieve_languages()	355
retrieve an array with all active languages from the database	
Language::reset_cache()	355
remove selected entries (per language+domain, per language, or all) from cache	
loginlib.php	659
/program/languages/fry/loginlib.php	
loginlib.php	680
/program/languages/hi/loginlib.php	
loginlib.php	862
/program/languages/tr/loginlib.php	
loginlib.php	843
/program/languages/sv/loginlib.php	
logger()	821
loginlib.php	871
/program/languages/ur/loginlib.php	
loginlib.php	891
/program/languages/vi/loginlib.php	
LICENSE	1341
loginlib.php	913
/program/languages/zh/loginlib.php	
loginlib.php	809
/program/languages/sk/loginlib.php	
loginlib.php	804
/program/languages/sh/loginlib.php	
loginlib.php	723
/program/languages/pap/loginlib.php	
loginlib.php	702

/program/languages/it/loginlib.php	689
loginlib.php	689
/program/languages/hu/loginlib.php	743
loginlib.php	743
/program/languages/pl/loginlib.php	756
loginlib.php	756
/program/languages/pt/loginlib.php	786
loginlib.php	786
/program/languages/ru/loginlib.php	774
loginlib.php	774
/program/languages/ro/loginlib.php	351
Language::get_current_language()	351
<i>determine the default language to use for translation of phrases</i>	
Language::get_active_language_names()	350
<i>return an array with active languages and language names</i>	
login_change_password()	85
<i>update the users database with a new (randomly salted) password and reset bypass mode to normal</i>	
LOGIN PROCEDURE SHOWLOGIN	82
<i>this only shows the login dialog</i>	
LOGIN PROCEDURE SEND LAISSEZ PASSER	82
<i>this is phase 1 of the 'forgot password' procedure</i>	
login_dialogdef()	86
<i>construct a standard dialog definition for a specific login procedure</i>	
login_failure_blacklist_address()	86
<i>add remote_addr to the blacklist for specified interval (in seconds)</i>	
login_failure_reset()	87
<i>deactivate all login failures/blacklisting scores for remote_addr</i>	
login_failure_increment()	87
<i>add 1 point to score for a particular IP-address and failed procedure, return the new score</i>	
login_failure_delay()	87
<i>delay execution of this script for a few seconds and blacklist the remote_addr during the delay</i>	
LOGIN PROCEDURE SEND BYPASS	82
<i>this is phase 2 of the 'forgot password' procedure</i>	
LOGIN PROCEDURE NORMAL	82
<i>this is the usual procedure for logging in</i>	
loginlib.php	80
<i>/program/lib/loginlib.php -- functions to handle user login/logout</i>	
language.class.php	79
<i>/program/lib/language.class.php - taking care of translations of messages</i>	
loginlib.php	19
<i>/program/languages/nl/loginlib.php - translated messages for login procedure and change password</i>	
LOGIN FAILURE DELAY SECONDS	82
<i>this is the number of seconds to delay responding after a login action fails (slow 'm down..)</i>	
LOGIN PROCEDURE BLACKLIST	82
<i>this is a pseudo procedure, used to record blacklisted IP-addresses</i>	
LOGIN PROCEDURE MESSAGE BOX	82
<i>this is a pseudo procedure, used to deliver some message to the user</i>	
LOGIN PROCEDURE CHANGE PASSWORD	82
<i>this is the procedure to change the user's password</i>	
login_is_blacklisted()	88
<i>find out if a remote address is blacklisted at this time</i>	

login_propagate_navigation()	88
<i>construct an action attribute propagating existing parameters</i>	
logger()	160
<i>a simple function to log information to the database 'for future reference'</i>	
lock_release_node()	160
<i>release lock on a node</i>	
lock_release()	159
<i>unlock a record that was previously successfully locked</i>	
Language	349
<i>Translations of messages in different languages</i>	
Language::\$default_domain	350
Language::\$phrases	350
Language::\$languages	350
lock_record_node()	159
<i>get record lock on a node</i>	
lock_record()	157
<i>put a (co-operative) lock on a record</i>	
login_send_laissez_passer()	90
<i>send a special one-time login code to the user via email</i>	
login_send_confirmation()	90
<i>send email to user confirming password change</i>	
login_send_bypass()	89
<i>send a new (temporary) password to the user via email</i>	
logview_download()	113
<i>download records from the log table in CSV format</i>	
logview_priority()	114
<i>helper routine to translate a numeric priority to human readable text</i>	
logview_show()	115
<i>show entries from log table in a neat HTML-table (possibly paginated)</i>	
logview_prune()	114
<i>ask for confirmation and/or maybe execute pruning of log messages table</i>	
loginlib.php	16
<i>/program/languages/en/loginlib.php - translated messages for login procedure and change password</i>	

M

mailpage_send_message()	1075
<i>actually send the visitor's message to the selected destination</i>	
mailpage_show_form()	1076
<i>display the contact form</i>	
MAILPAGE_REFERENCE	1075
mailpage_view.php	1075
<i>/program/modules/mailpage/mailpage_view.php - interface to the view-part of the mailpage module</i>	
mailpage_search()	1074
<i>search this module's content in selected nodes for keywords in \$qwords</i>	
mailpage_show_preview()	1076
<i>show a preview of the message to the visitor</i>	
mailpage_show_thankyou()	1077
<i>thank the visitor for the message and show a text copy too</i>	
mailpage_view_get_dialogdef()	1078

<i>construct a dialog definition for the visitor's mail form</i>	1080
mypage.php	
<i>/program/modules/mypage/languages/en/mypage.php - translated messages for module (English)</i>	
mailpage view_get_config()	1078
<i>retrieve all configuration data for this mailpage</i>	
mailpage view_dialog_validate()	1077
<i>validate the data entered by the visitor</i>	
mailpage view()	1077
<i>display the content of the mailpage linked to node \$node_id</i>	
mailpage search.php	1074
<i>/program/modules/mailpage/mailpage_search.php - interface to the search-part of this module</i>	
mailpage manifest.php	1073
<i>/program/modules/mailpage/mailpage_manifest.php - description of the mailpage module</i>	
mailpage show_edit_destinations()	1066
<i>present the user with a dialog for mailpage destination addresses at node \$node_id</i>	
mailpage show_edit_destinations_overview()	1068
<i>show a screen with a links to add/edit/delete destination addresses</i>	
mailpage show_edit_configuration()	1066
<i>present the user with the main configuration dialog for mailpage at node \$node_id</i>	
mailpage show_edit()	1065
<i>dispatcher for presenting the user with one of the mailpage dialogs</i>	
mailpage show_content()	1065
<i>show the main content screen (in fact documentation only)</i>	
mailpage cron.php	1069
<i>/program/modules/mailpage/mailpage_cron.php - interface to the cron-part of the mailpage module</i>	
mailpage cron()	1069
<i>routine that is called periodically by cron</i>	
mailpage uninstall()	1071
<i>uninstall the module</i>	
mailpage upgrade()	1071
<i>upgrade the module</i>	
mailpage install()	1071
<i>install the module</i>	
mailpage demodata()	1070
<i>add demonstration data to the system</i>	
mailpage install.php	1070
<i>/program/modules/mailpage/mailpage_install.php - installer of the mailpage module</i>	
mypage.php	1081
<i>/program/modules/mypage/languages/nl/mypage.php - translated messages for module (Dutch)</i>	
mypage admin.php	1082
<i>/program/modules/mypage/mypage_admin.php - management interface for mypage-module</i>	
mypage password_get_dialogdef()	1092
<i>construct the edit dialog for the password of the current user</i>	
mypage profile_dialog_validate()	1092
<i>validate entries in user profile dialogue</i>	
mypage password_dialog_validate()	1092
<i>validate entries in user password dialogue</i>	
mypage authorisation()	1091
<i>check the authorisation for a submitted dialogue</i>	
mypage view.php	1091

<i>/program/modules/mypage/mypage_view.php - interface to the view-part of the mypage module</i>	
mypage_profile_get_dialogdef()	1092
<i>construct the edit dialog for the profile of the current user</i>	
mypage_profile_save()	1093
<i>save modified user profile in database and keep a copy in core</i>	
mypage_view_password()	1094
<i>show/process password dialogue</i>	
mypage_view_profile()	1095
<i>show/process profile dialogue</i>	
mypage_view_login()	1094
<i>show a login dialog</i>	
mypage_view_home()	1094
<i>show links to profile, admin.php and available areas</i>	
mypage_view()	1093
<i>display the content mypage linked to node \$node_id</i>	
mypage_search()	1090
<i>search this module's content in selected nodes for keywords in \$qwords</i>	
mypage_search.php	1090
<i>/program/modules/mypage/mypage_search.php - interface to the search-part of this module</i>	
mypage_show_edit()	1084
<i>present the user with a dialog to modify the mypage that is connected to node \$node_id</i>	
mypage_cron.php	1085
<i>/program/modules/mypage/mypage_cron.php - interface to the cron-part of the mypage module</i>	
mypage_save()	1083
<i>save the modified content data of this module linked to node \$node_id</i>	
mypage_disconnect()	1082
<i>disconnect this module from a node</i>	
mypage_connect()	1082
<i>connect this module to a node</i>	
mypage_cron()	1085
<i>routine that is called periodically by cron</i>	
mypage_install.php	1086
<i>/program/modules/mypage/mypage_install.php - installer of the mypage module</i>	
mypage_upgrade()	1087
<i>upgrade the module</i>	
mypage_manifest.php	1089
<i>/program/modules/mypage/mypage_manifest.php - description of the mypage module</i>	
mypage_uninstall()	1087
<i>uninstall the module</i>	
mypage_install()	1087
<i>install the module</i>	
mypage_demodata()	1086
<i>add demonstration data to the system</i>	
mailpage_save_destinations()	1064
<i>process the POSTed destination address data</i>	
mailpage_save_configuration()	1064
<i>save the modified configuration data of this mailpage linked to node \$node_id</i>	
main_admin.php	175
<i>/program/main_admin.php - workhorse for site maintenance</i>	
main_admin()	179
<i>main program for site maintenance</i>	

magic_unquote()	161
<i>this circumvents the 'magic' in magic_quotes_gpc() by conditionally stripping slashes</i>	
MODULE_NAME_DEFAULT	100
<i>Default initial module of a new page (see get_dialogdef_add_node())</i>	
modulemanager_show_menu()	99
<i>display the module manager menu</i>	
main_cron.php	181
<i>/program/main_cron.php - take care of recurring jobs</i>	
main_cron()	182
main_index()	191
<i>main program for visitors</i>	
module_load_view()	191
<i>load the visitor/view interface of a module in core</i>	
main_index.php	189
<i>/program/main_index.php - workhorse for visitor interface</i>	
main_file()	184
<i>main program for serving files</i>	
main_file.php	183
<i>/program/main_file.php - workhorse for serving files</i>	
modulemanager_show_intro()	98
<i>display an introductory text for the module manager</i>	
modulemanager_process()	98
<i>handle the editing/saving of the main configuration information</i>	
mime_type_by_ext()	69
<i>lookup mimetype in table, using extension as key</i>	
mime_type_ext_match()	70
<i>verify the combination of filename extension and mimetype</i>	
mime_ext_by_type()	69
<i>lookup extension in table, using mimetype as key</i>	
mysqli.class.php	41
<i>/program/lib/database/mysqli.class.php - access to mysql via database class</i>	
mysql.class.php	40
<i>/program/lib/database/mysql.class.php - access to mysql via database class</i>	
MINIMUM_PASSWORD_DIGITS	82
<i>this is the hardcoded minimal number of digits in a new password</i>	
MINIMUM_PASSWORD_LENGTH	83
<i>this hardcoded minimal length is enforced whenever a user wants to change her password</i>	
modulemanager_cmp()	97
<i>compare two arrays by the title member (for sorting modules)</i>	
modulemanager_get_modules()	97
<i>retrieve a list of modules that should appear in the module manager</i>	
modulemanagerlib.php	97
<i>/program/lib/modulemanagerlib.php - modulemanager</i>	
MINIMUM_PASSWORD_UPPERCASE	83
<i>this is the hardcoded minimal number of upper case characters in a new password</i>	
MINIMUM_PASSWORD_LOWERCASE	83
<i>this is the hardcoded minimal number of lower case characters in a new password</i>	
module_view()	192
<i>call the routine that generates the view (content) of module \$module_id</i>	
manual.php	194
<i>/program/manual.php - a kickstarter for the documentation</i>	
mailpage_connect()	1060
<i>connect this module to a node</i>	

mailpage_dialog_validate_address()	1061
<i>validate the data entered by the user</i>	
mailpage_admin.php	1060
<i>/program/modules/mailpage/mailpage_admin.php - management interface for mailpage-module</i>	
mailpage.php	1059
<i>/program/modules/mailpage/languages/nl/mailpage.php - translated messages for module (Dutch)</i>	
mailpage.php	1058
<i>/program/modules/mailpage/languages/en/mailpage.php - translated messages for module (English)</i>	
mailpage_disconnect()	1061
<i>disconnect this module from a node</i>	
mailpage_get_addresses()	1061
<i>retrieve current list of addresses in an array (could be empty)</i>	
mailpage_get_icon_delete()	1063
<i>construct a delete button for deletion of a destination address</i>	
mailpage_save()	1063
<i>dispatcher for save routines</i>	
mailpage_get_dialogdef_delete()	1062
<i>construct a dialog definition for deletion of a destination address</i>	
mailpage_get_dialogdef_config()	1062
<i>construct a dialog definition for the main mailpage configuration</i>	
mailpage_get_dialogdef_address()	1062
<i>construct a dialog definition for a mailpage destination address (add/edit)</i>	
mailpage_tabledefs.php	1057
<i>/program/modules/mailpage/install/mailpage_tabledefs.php - data definition for module</i>	
mailpage.php	899
<i>/program/modules/mailpage/languages/vi/mailpage.php</i>	
mailpage.php	666
<i>/program/modules/mailpage/languages/fry/mailpage.php</i>	
mailpage.php	707
<i>/program/modules/mailpage/languages/it/mailpage.php</i>	
mailpage.php	644
<i>/program/modules/mailpage/languages/fr/mailpage.php</i>	
mailpage.php	556
<i>/program/modules/mailpage/languages/de/mailpage.php</i>	
mailpage.php	523
<i>/program/modules/mailpage/languages/bg/mailpage.php</i>	
mailpage.php	729
<i>/program/modules/mailpage/languages/pap/mailpage.php</i>	
mailpage.php	762
<i>/program/modules/mailpage/languages/pt/mailpage.php</i>	
mailpage.php	877
<i>/program/modules/mailpage/languages/ur/mailpage.php</i>	
mailpage.php	849
<i>/program/modules/mailpage/languages/sv/mailpage.php</i>	
main()	821
mailpage.php	792
<i>/program/modules/mailpage/languages/ru/mailpage.php</i>	
MAXIMUM_ITERATIONS	10
<i>This global constant defines the maximum number of iterations in database loops (prevent circular reference)</i>	

N

NEWSLETTER SUBSCRIPTION BLACKLIST	1130
NEWSLETTER SUBSCRIPTION CONFIRMED	1130
NEWSLETTER SUBSCRIPTION NEW	1130
NEWSLETTER SUBSCRIPTION APPROVED	1130
NEWSLETTER FULLMAIL SITE	1130
NEWSLETTER CONTRIBUTION POLICY YES	1130
NEWSLETTER FULLMAIL ONLY	1130
NEWSLETTER SUBSCRIPTION POLICY NONE	1130
NEWSLETTER SUBSCRIPTION POLICY ONE	1130
newsletter consolidate subscribers()	1131
<i>remove obsolete subscriber data from the database</i>	
newsletter dialog validate()	1131
<i>validate the data entered by the (un)subscriber</i>	
newsletter get_emails()	1132
<i>create a neat array with email-addresses from a (possibly messy) text</i>	
newsletter config()	1130
<i>retrieve the newsletter configuration</i>	
NEWSLETTER UNPUBLISHED	1130
NEWSLETTER SUBSCRIPTION POLICY TWO	1130
NEWSLETTER SUBSCRIPTION SKIP	1130
NEWSLETTER CONTRIBUTION POLICY USERS	1130
NEWSLETTER CONTRIBUTION POLICY NO	1130
NEWSLETTER CHORE DOWNLOAD	1130
NEWSLETTER CHORE EDIT	1130
NEWSLETTER CHORE LIST	1130
NEWSLETTER CHORE DOWN	1130
NEWSLETTER CHORE DESELECT	1130
NEWSLETTER CHORE ADD	1130
NEWSLETTER CHORE DELETE	1130
NEWSLETTER CHORE PREVIEW	1130
NEWSLETTER CHORE SELECT	1130
NEWSLETTER CONFIRM DEL	1130
NEWSLETTER CONFIRM LEN	1130
NEWSLETTER CONFIRM ADD	1130
NEWSLETTER CHORE UPLOAD	1130
NEWSLETTER CHORE TESTMAIL	1130
NEWSLETTER CHORE UP	1130
newsletter queue run()	1132
<i>attempt to send out as many queued newsletters as possible</i>	
newsletter cron.php	1134
<i>/program/modules/newsletter/newsletter_cron.php - interface to the cron-part of the newsletter module</i>	
newsletter view contribution get_dialogdef()	1144
<i>construct a dialog definition for the visitor's contribution form</i>	
newsletter view contribution preview()	1145
<i>show a preview of the message to the visitor</i>	
newsletter view contribution send()	1145
<i>add the contribution to the list of unselected articles and mail the admins</i>	
newsletter view contribution form()	1144
<i>display the contribution form</i>	
newsletter view contribution()	1143

<i>handle user contribution of articles</i>	
newsletter_view_confirm_get_dialogdef()	1143
<i>construct a dialog definition for confirm code</i>	
newsletter_view_confirm_process_code()	1143
<i>process a code confirming a subscribe or unsubscribe action</i>	
newsletter_view_contribution_thankyou()	1146
<i>thank the visitor for the article and show a text copy too</i>	
newsletter_view_home()	1146
<i>display the default page, possibly with the latest newsletter embedded</i>	
newsletter_view_unsubscribe()	1148
<i>handle unsubscriptions (phase 1)</i>	
newsletter_view_unsubscribe_get_dialogdef()	1148
<i>construct a dialog definition for unsubscribing</i>	
newsletter_view_unsubscribe_send_code()	1148
<i>send a quasi-random unsubscribe confirmation code to \$email</i>	
newsletter_view_subscribe_send_code()	1147
<i>send a quasi-random subscribe confirmation code to \$email</i>	
newsletter_view_subscribe_get_dialogdef()	1147
<i>construct a dialog definition for subscribing</i>	
newsletter_view_issue()	1146
<i>show a 'stand-alone' version of the newsletter</i>	
newsletter_view_subscribe()	1147
<i>show and process a subscription request (phase 1)</i>	
newsletter_view_confirm()	1143
<i>handle the confirmation code for subscribing/unsubscribing</i>	
newsletter_view_archive()	1142
<i>display two levels of newsletter issues: volumes+number or numbers+toc</i>	
newsletter_uninstall()	1137
<i>uninstall the module</i>	
newsletter_upgrade()	1137
<i>upgrade the module</i>	
newsletter_manifest.php	1138
<i>/program/modules/newsletter/newsletter_manifest.php - description of the newsletter module</i>	
newsletter_install()	1136
<i>install the module</i>	
newsletter_demodata()	1135
<i>add demonstration data to the system</i>	
newsletter_cron()	1134
<i>routine that is called periodically by cron</i>	
newsletter_install.php	1135
<i>/program/modules/newsletter/newsletter_install.php - installer of the newsletter module</i>	
newsletter_search.php	1139
<i>/program/modules/newsletter/newsletter_search.php - interface to the search-part of this module</i>	
newsletter_search()	1139
<i>search this module's content in selected nodes for keywords in \$qwords</i>	
newsletter_view()	1141
<i>display the content of the newsletter linked to node \$node_id</i>	
newsletter_view_add_navbar()	1142
<i>add a navigation bar to the output</i>	
NEWSLETTER REFERENCE	1141
NEWSLETTER_DIRNAME	1141
newsletter_search_results()	1140

count the number of hits in \$table	
newsletter_view.php	1141
/program/modules/newsletter/newsletter_view.php - interface to the view-part of the newsletter module	
NEWSLETTER_ALERT_SITE	1130
newsletter_common.php	1130
/program/modules/newsletter/newsletter_common.php - code shared between admin and view	
newsletter_compose_show_list_article()	1107
output various icons and links for manipulating articles	
newsletter_compose_show_preview()	1107
newsletter_compose_show_testmail()	1108
send an example mail of the current newsletter to the sender address	
newsletter_compose_show_list()	1106
create the main newsletter compose screen	
newsletter_compose_show_edit()	1105
show dialog for editing (or adding) an article	
newsletter_compose_show_add()	1104
show the dialog for adding a new article	
newsletter_compose_show_delete()	1105
show a delete confirmation dialog	
newsletter_compose_toggle()	1108
select or deselect an article	
newsletter_configuration_dialog_validate()	1108
validation of configuration dialog + rewriting the list of administrator email	
newsletter_content_show()	1110
show the main content screen (in fact documentation only)	
newsletter_disconnect()	1111
disconnect this module from a node	
newsletter_download_csv()	1111
send a CSV file of subscribers to the browser	
newsletter_connect()	1110
connect this module to a node	
newsletter_configuration_show()	1110
show a simple edit screen for the newsletter configuration	
newsletter_configuration_get_dialogdef()	1109
construct a dialog definition for the main newsletter configuration	
newsletter_configuration_save()	1109
validate and save the modified newsletter configuration linked to node \$node_id	
newsletter_compose_show()	1104
dispatcher for displaying newsletter composition dialogs	
newsletter_compose_save()	1104
handle saving & deleting of articles	
nl_manifest.php	718
/program/languages/nl/nl_manifest.php - description of the Dutch translation	
newsletter_tabledefs.php	1098
/program/modules/newsletter/install/newsletter_tabledefs.php - data definition for module	
newsletter.php	1099
/program/modules/newsletter/languages/en/newsletter.php - translated messages for module (English)	
non_admin_redirect_and_exit()	180
tell non-admin-user access denied and exit	
NODE_VISIBILIY_VISIBLE	100
Initial visibility of a new node: visible	

NODE VISIBILIY EMBARGO	100
<i>Initial visibility of a new node: under embargo</i>	
NODE VISIBILIY HIDDEN	100
<i>Initial visibility of a new node: hidden</i>	
newsletter.php	1100
<i>/program/modules/newsletter/languages/en/newsletter.php - translated messages for module (Dutch)</i>	
newsletter_admin.php	1101
<i>/program/modules/newsletter/newsletter_admin.php - management interface for newsletter-module</i>	
newsletter_compose_move()	1103
<i>swap two articles in the list of ((de)selected) articles</i>	
newsletter_compose_new_sort_order()	1103
<i>calculate a new sort order based on the current values of available articles in this newsletter</i>	
newsletter_compose_get_dialogdef_delete()	1103
<i>construct a dialog definition for deletion of an article</i>	
newsletter_compose_get_dialogdef()	1102
<i>construct a dialog definition for a newsletter article (add/edit)</i>	
newsletter_add_related()	1101
<i>add the file in path as a related attachment to the email being built</i>	
newsletter_body2text()	1101
<i>convert the body of an HTML-document to text</i>	
newsletter_get_html()	1112
<i>construct the HTML-version of the newsletter in parts</i>	
newsletter_get_html_templates()	1113
<i>construct the configurable part of the HTML-version of the newsletter, with parameters</i>	
newsletter_subscriptions_show_list()	1122
<i>create the main newsletter subscriptions overview screen</i>	
newsletter_subscriptions_table()	1123
<i>pretty-print a list of subscribers including sort options in the HTML table header</i>	
newsletter_upload_csv()	1124
<i>handle subscribers upload</i>	
newsletter_subscriptions_show_edit()	1122
<i>show dialog for editing (or adding) a subscriber</i>	
newsletter_subscriptions_show_delete()	1121
<i>show a delete confirmation dialog</i>	
newsletter_subscriptions_show()	1120
<i>dispatcher for displaying newsletter subscriptions dialogs</i>	
newsletter_subscriptions_show_add()	1121
<i>show the dialog for adding a new subscriber</i>	
newsletter_upload_csv_extract()	1124
<i>helper routine to extract usable data from a single CSV record</i>	
newsletter_upload_csv_get_dialogdef()	1125
<i>construct a dialog for specifying CSV upload</i>	
newsletter_urls2cids()	1128
<i>search through \$html add selected files to email and replace url with cid</i>	
newsletter_volume_number_available()	1128
<i>determine whether volume + number already exists in this newsletter</i>	
newsletter_upload_entries_store()	1127
<i>store the subscribers data in the database</i>	
newsletter_upload_entries_get_dialogdef()	1126
<i>construct a dialog definition, maybe filled with data from \$uploads</i>	
newsletter_upload_csv_load()	1125

<i>message the submitted data and return a neat array with subscriber information</i>	1125
<u>newsletter_upload_entries()</u>	1125
<i>handle phase two of subscriber upload</i>	
<u>newsletter_subscriptions_save()</u>	1120
<i>handle saving & deleting of subscribers</i>	
<u>newsletter_subscriptions_get_dialogdef_delete()</u>	1119
<i>construct a dialog definition for deletion of a subscriber</i>	
<u>newsletter_publish_show()</u>	1115
<i>show the newsletter publication confirmation screen for the current newsletter</i>	
<u>newsletter_publish_show_confirm()</u>	1115
<i>show a confirmation dialog for the publisher</i>	
<u>newsletter_publish_show_preview()</u>	1116
<u>newsletter_publish_save()</u>	1114
<i>publish the prepared newsletter</i>	
<u>newsletter_publish_prepare_issue()</u>	1114
<i>prepare the current newsletter for publication in the issues table</i>	
<u>newsletter_publish_dialog_validate()</u>	1114
<i>validation of publication confirmation dialog checks checked checkbox</i>	
<u>newsletter_publish_get_dialogdef()</u>	1114
<i>construct the dialog definition for confirmation of newsletter publication</i>	
<u>newsletter_publish_show_testmail()</u>	1116
<i>send final example mail of the current newsletter from database to the publishers address</i>	
<u>newsletter_queue_get_dialogdef()</u>	1117
<i>construct an OK-button and a Cancel-button for the queue overview dialog</i>	
<u>newsletter_strip8859_1()</u>	1119
<i>transliterate characters 128-255 to something 'ascii-like'</i>	
<u>newsletter_subscriptions_get_dialogdef()</u>	1119
<i>construct a dialog definition for a subscriber (add/edit)</i>	
<u>newsletter_show_edit()</u>	1118
<i>dispatcher for showing various dialog screens</i>	
<u>newsletter_save()</u>	1118
<i>dispatcher for save routines</i>	
<u>newsletter_queue_save()</u>	1117
<i>process a few entries in the queue (if any)</i>	
<u>newsletter_queue_show()</u>	1117
<i>display a table with pending newsletter emails</i>	
<u>NODE_VISIBILIY_DEFAULT</u>	100
<i>Default initial visibility of a new node (see <u>get_dialogdef_add_node()</u>)</i>	

P

<u>PageManager::permission_edit_node_content()</u>	376
<i>does the user have the privilege to edit node content?</i>	
<u>PageManager::permission_set_default()</u>	376
<i>does the user have the privilege to make node \$node_id the default?</i>	
<u>PageManager::permission_edit_node()</u>	375
<i>does the user have the privilege to edit node properties?</i>	
<u>PageManager::permission_delete_node()</u>	375
<i>does the user have the privilege to delete a node from the area?</i>	
<u>PageManager::permission_add_node()</u>	374
<i>does the user have the privilege to add a node to an area or a section?</i>	
<u>PageManager::save_node()</u>	377

workhorse routing for saving modified node data to the database	
PageManager::save_node_new_area_mass_move()	378
workhorse routine for moving a complete subtree to another area	
PageManager::show_dialog_force_unlock()	380
show a dialog to the user offering to forcefully unlock a node	
PageManager::show_edit_menu()	381
construct a clickable list of edit variants (basic, advanced and maybe content)	
PageManager::show_dialog_delete_node_confirm()	380
display a list of 1 or more nodes to delete and ask user for confirmation of delete	
PageManager::show_area_menu()	379
construct a clickable list of available areas for the current user	
PageManager::section_is_open()	379
shorthand for determing whether a section is opened or closed	
PageManager::permission_add_any_node()	374
does the user have the privilege to add a node, any node to an area?	
PageManager::node_has_grandchildren()	374
shorthand to determine whether the number of levels below section \$node_id is greater than one	
PageManager::lock_records_subtree()	369
attempt to lock all node records in a subtree	
PageManager::message_from_lockinfo()	370
construct a readable message from the lockinfo array	
PageManager::get_options_sort_order()	368
generate a list of siblings in a particular (sub)section used to select/change sort order via a list box	
PageManager::get_options_parents_walk()	368
workhorse for construction an options list of possible parent sections	
PageManager::get_options_parents()	367
construct an options list of possible parent sections	
PageManager::module_connect()	370
inform module \$module_id that from now on it will be linked to page \$node_id	
PageManager::module_disconnect()	371
inform module \$module_id that it is no longer linked to page \$node_id	
PageManager::node_full_name()	373
shorthand for constructing a readable page/section name with id, name and title	
PageManager::module_show_edit()	373
show a dialog for editing the content of module \$module_id linked to page \$node_id	
PageManager::module_save()	372
(maybe) save the modified content of module \$module_id connected to page \$node_id	
PageManager::module_load_admin()	371
load the admin interface of a module in core	
PageManager::show_tree()	381
create a tree-like list of nodes in the content area of \$this->output	
PageManager::show_treeview_buttons()	382
show one or two clickable links to change the view of the tree	
pap_manifest.php	724
/program/languages/pap/pap_manifest.php - description of the Papiamento translation	
pl_manifest.php	744
/program/languages/pl/pl_manifest.php - description of the Polish translation	
PROJECT_SITE	456
PageManager::task_treeview_set()	391
this sets the tree view to the specified mode	
PageManager::task_treeview()	390

<i>maybe change the current area and then show the tree and the menu for the current area</i>	
pt_manifest.php	757
<i>/program/languages/pt/pt_manifest.php - description of the Portugese translation</i>	
print_checker_results()	816
print_words_elem()	816
print_textinputs_var()	816
print_textindex_decl()	816
print_suggs_elem()	816
PageManager::task_subtree_expand()	390
<i>open the selected section and perhaps change the view mode</i>	
PageManager::task_subtree_collapse()	389
<i>close the selected section and perhaps change the view mode</i>	
PageManager::task_node_delete()	384
<i>delete one or more nodes from an area after user confirmation</i>	
PageManager::task_node_edit()	385
<i>display a dialog where the user can edit basic or advanced properties of a node</i>	
PageManager::task_node_add()	384
<i>display a dialog to add a new page or section to the current area</i>	
PageManager::task_force_unlock()	384
<i>forcefully obtain a lock on a node and release it immediately</i>	
PageManager::show_tree_walk()	383
<i>display the specified node, optionally all subtrees, and subsequently all siblings</i>	
PageManager::task_node_edit_content()	386
<i>display a dialog where the user can edit the contents of a node via a module</i>	
PageManager::task_page_preview()	386
<i>preview a page that is maybe still under embargo/already expired</i>	
PageManager::task_set_default()	389
<i>make the selected node the default for this level</i>	
PageManager::task_save_node()	389
PageManager::task_save_newnode()	388
<i>save a newly added node to the database</i>	
PageManager::task_save_content()	388
PageManager::get_options_modules()	367
<i>fetch a list of available modules for inclusion on a page</i>	
PageManager::get_options_area()	366
<i>generate a list of areas for use in a dropdown list (for moving a node to another area)</i>	
PERMISSION_NODE_ADD_SECTION	137
PERMISSION_NODE_DROP_CONTENT	137
PERMISSION_NODE_ADD_PAGE	137
PERMISSION_NODE_ADD_CONTENT	137
PERMISSION_AREA_EDIT_AREA	137
PERMISSION_NODE_DROP_PAGE	137
PERMISSION_NODE_DROP_SECTION	137
PERMISSION_SITE_ADD_AREA	137
PERMISSION_SITE_DROP_AREA	137
PERMISSION_NODE_EDIT_SECTION	137
PERMISSION_NODE_EDIT_PAGE	137
PERMISSION_NODE_EDIT_CONTENT	137
PERMISSION_AREA_DROP_SECTION	137
PERMISSION_AREA_DROP_PAGE	137
PARAM_SORT	71
password_hash_check()	91
<i>check equivalency of salt+password against hash</i>	

PARAM_PATH	71
PARAM_FILENAMES	71
<i>This constant is used to construct the fieldname counting the number of files to delete</i>	
PARAM_FILENAME	71
<i>This constant is used to construct the fieldname used for deleting files</i>	
password_salt()	92
<i>generate a quasi random string to salt the password hash</i>	
pagemanager.class.php	100
<i>/program/lib/pagemanager.class.php - pagemanager</i>	
PERMISSION_AREA_ADD_SECTION	137
PERMISSION_AREA_ADD_PAGE	137
PARAM_TREEVIEW	100
PARAM_SUBMENU_OPTION	100
PERMISSION_SITE_EDIT_SITE	137
performance_get_queries()	161
<i>return the number of database queries that was executed</i>	
PageManager::get_icon_delete()	362
<i>construct a clickable icon to delete this node (and underlying nodes too)</i>	
PageManager::get_icon_edit()	363
<i>construct a clickable icon to edit this node</i>	
PageManager::get_dialog_data_node()	362
<i>fill the node dialog with data from the database</i>	
PageManager::get_dialogdef_force_unlock()	362
<i>construct a dialog definition to show [OK] and [Cancel]</i>	
PageManager::get_dialogdef_edit_node()	361
<i>construct a dialog definition for editing basic properties of an existing node (page or section)</i>	
PageManager::get_icon_home()	363
<i>construct a clickable icon to set the home page/section on this tree level</i>	
PageManager::get_icon_invisibility()	364
<i>construct a clickable icon to edit the advanced properties of this node</i>	
PageManager::get_node_id_and_ancestors()	366
<i>get an array with all ids of ancestors of node_id and node_id itself</i>	
PageManager::get_link_node_edit()	365
<i>construct a clickable link to edit this page OR open/close this section</i>	
PageManager::get_icon_section()	365
<i>construct a clickable icon to open/close this node</i>	
PageManager::get_icon_page_preview()	364
<i>construct a clickable icon to preview this node</i>	
PageManager::get_dialogdef_edit_advanced_node()	361
<i>construct a dialog definition for editing advanced properties of a node (page or section)</i>	
PageManager::get_dialogdef_add_node()	360
<i>construct a dialog definition for adding a node (page or section)</i>	
PageManager::\$area_id	356
PageManager::\$output	356
PageManager::\$areas	356
PageManager	356
<i>Page Manager</i>	
performance_get_seconds()	162
<i>return the script execution time</i>	
PageManager::\$tree	357
PageManager::build_cached_tree()	357
<i>construct \$this->tree for future reference</i>	
PageManager::delete_node()	360

<i>workhorse routine for deleting a node, including children</i>	
PageManager::calc_home_id()	360
<i>calculate the current default node on this level</i>	
PageManager::calculate_updated_sort_order()	358
<i>calculate an updated sort order and also make space in the section for moving the node around</i>	
PageManager::calculate_new_sort_order()	358
<i>calculate a new sort order and at the same time make room for a node</i>	
process_task_site()	29
<i>handle the editing/saving of the main configuration information</i>	

Q

queue_area_node_alert()	162
<i>add a message to message queue of 0 or more alerts</i>	
quoted_printable()	164
<i>convert string \$s from native format to quoted printable (RFC2045)</i>	
quasi_random_string()	162
<i>generate a string with quasi-random characters</i>	
QUASI_RANDOM_HEXDIGITS	144
QUASI_RANDOM_DIGITS_UPPER	144
QUASI_RANDOM_DIGITS_UPPER_LOWER	144
QUASI_RANDOM_DIGITS	144

R

redirect_demodata()	1166
<i>add demonstration data to the system</i>	
redirect_install.php	1166
<i>/program/modules/redirect/redirect_install.php - installer of the redirect module</i>	
redirect_cron()	1165
<i>routine that is called periodically by cron</i>	
redirect_cron.php	1165
<i>/program/modules/redirect/redirect_cron.php - interface to the cron-part of the redirect module</i>	
redirect_install()	1167
<i>install the module</i>	
redirect_uninstall()	1167
<i>uninstall the module</i>	
redirect_search.php	1170
<i>/program/modules/redirect/redirect_search.php - interface to the search-part of this module</i>	
redirect_manifest.php	1169
<i>/program/modules/redirect/redirect_manifest.php - description of the redirect module</i>	
redirect_upgrade()	1167
<i>upgrade the module</i>	
redirect_show_edit()	1163
<i>present the user with a dialog to modify the redirect that is connected to node \$node_id</i>	
redirect_save()	1162
<i>save the modified content data of this module linked to node \$node_id</i>	
redirect.php	1159
<i>/program/modules/redirect/languages/tr/redirect.php</i>	
redirect.php	1158

/program/modules/redirect/languages/pl/redirect.php	1157
redirect.php	1157
/program/modules/redirect/languages/nl/redirect.php - translated messages for module (Dutch)	1156
redirect.php	1156
/program/modules/redirect/languages/hu/redirect.php	1160
redirect.php	1160
/program/modules/redirect/languages/zh/redirect.php	1161
redirect_admin.php	1161
/program/modules/redirect/redirect_admin.php - management interface for redirect-module	1162
redirect_get_dialogdef()	1162
construct a dialog definition for the redirect value	
redirect_disconnect()	1161
disconnect this module from a node	
redirect_connect()	1161
connect this module to a node	
redirect_search()	1170
search this module's content in selected nodes for keywords in \$qwords	
redirect_view.php	1171
/program/modules/redirect/redirect_view.php - interface to the view-part of the redirect module	1294
ruta_demodata()	1294
add demonstration data to the system	
ruta_install.php	1294
/program/themes/ruta/ruta_install.php -- installer for the ruta theme	1293
ruta.php	1293
/program/themes/ruta/languages/nl/ruta.php - translated messages for theme (Nederlands)	1292
ruta.php	1292
/program/themes/ruta/languages/en/ruta.php - translated messages for theme (English)	1295
ruta_install()	1295
install the theme	
ruta_uninstall()	1295
uninstall the theme	
README	1341
ruta_manifest.php	1297
/program/themes/ruta/ruta_manifest.php - description of the ruta theme	1296
ruta_upgrade()	1296
upgrade the theme	
rosalina_manifest.php	1283
/program/themes/rosalina/rosalina_manifest.php - description of the rosalina theme	1281
rosalina_upgrade()	1281
upgrade the theme	
rosalina.class.php	1278
/program/themes/rosalina/rosalina.class.php - a theme with HV Menu (Javascript-based)	1277
rosalina.php	1277
/program/themes/rosalina/languages/nl/rosalina.php - translated messages for theme (Nederlands)	1276
rosalina.php	1276
/program/themes/rosalina/languages/en/rosalina.php - translated messages for theme (English)	1171
redirect_view()	1171
display the content of the redirect linked to node \$node_id	
rosalina_install.php	1279
/program/themes/rosalina/rosalina_install.php -- installer of the rosalina theme	1279
rosalina_demodata()	1279

<i>add demonstration data to the system</i>	1281
rosalina_uninstall()	1281
<i>uninstall the theme</i>	
rosalina_install()	1280
<i>install the theme</i>	
rosalina_get_properties()	1280
<i>construct a list of default properties for this theme</i>	
redirect.php	1155
<i>/program/modules/redirect/languages/fa/redirect.php</i>	
redirect.php	1154
<i>/program/modules/redirect/languages/es/redirect.php</i>	
redirect.php	645
<i>/program/modules/redirect/languages/fr/redirect.php</i>	
rosalina.php	627
<i>/program/themes/rosalina/languages/fi/rosalina.php</i>	
rosalina.php	614
<i>/program/themes/rosalina/languages/fa/rosalina.php</i>	
rosalina.php	600
<i>/program/themes/rosalina/languages/es/rosalina.php</i>	
rosalina.php	651
<i>/program/themes/rosalina/languages/fr/rosalina.php</i>	
redirect.php	667
<i>/program/modules/redirect/languages/fry/redirect.php</i>	
redirect.php	708
<i>/program/modules/redirect/languages/it/redirect.php</i>	
rosalina.php	695
<i>/program/themes/rosalina/languages/hu/rosalina.php</i>	
rosalina.php	673
<i>/program/themes/rosalina/languages/fry/rosalina.php</i>	
rosalina.php	582
<i>/program/themes/rosalina/languages/el/rosalina.php</i>	
redirect.php	578
<i>/program/modules/redirect/languages/el/redirect.php</i>	
rosalina.php	510
<i>/program/themes/rosalina/languages/ar/rosalina.php</i>	
rfc1123date()	186
<i>generate an RFC1123-compliant date/time stamp</i>	
readfile_chunked()	186
<i>send a file to the visitor's browser in chunks</i>	
replace_crlf()	166
<i>unfold a possible multiline string</i>	
redirect.php	524
<i>/program/modules/redirect/languages/bg/redirect.php</i>	
rosalina.php	530
<i>/program/themes/rosalina/languages/bg/rosalina.php</i>	
rosalina.php	563
<i>/program/themes/rosalina/languages/de/rosalina.php</i>	
redirect.php	557
<i>/program/modules/redirect/languages/de/redirect.php</i>	
rosalina.php	544
<i>/program/themes/rosalina/languages/da/rosalina.php</i>	
rosalina.php	714
<i>/program/themes/rosalina/languages/it/rosalina.php</i>	

redirect.php	730
/program/modules/redirect/languages/pap/redirect.php	
redirect.php	900
/program/modules/redirect/languages/vi/redirect.php	
rosalina.php	884
/program/themes/rosalina/languages/ur/rosalina.php	
redirect.php	878
/program/modules/redirect/languages/ur/redirect.php	
rosalina.php	856
/program/themes/rosalina/languages/sv/rosalina.php	
rosalina.php	906
/program/themes/rosalina/languages/vi/rosalina.php	
rosalina.php	920
/program/themes/rosalina/languages/zh/rosalina.php	
redirect.php	1153
/program/modules/redirect/languages/en/redirect.php - translated messages for module (English)	
redirect.php	1152
/program/modules/redirect/languages/da/redirect.php	
redirect.php	1151
/program/modules/redirect/languages/ar/redirect.php	
redirect.php	850
/program/modules/redirect/languages/sv/redirect.php	
read_config()	821
rosalina.php	769
/program/themes/rosalina/languages/pt/rosalina.php	
redirect.php	763
/program/modules/redirect/languages/pt/redirect.php	
rosalina.php	750
/program/themes/rosalina/languages/pl/rosalina.php	
rosalina.php	736
/program/themes/rosalina/languages/pap/rosalina.php	
ro_manifest.php	775
/program/languages/ro/ro_manifest.php - description of the Romanian translation	
rosalina.php	781
/program/themes/rosalina/languages/ro/rosalina.php	
rosalina.php	799
/program/themes/rosalina/languages/ru/rosalina.php	
redirect.php	793
/program/modules/redirect/languages/ru/redirect.php	
ru_manifest.php	787
/program/languages/ru/ru_manifest.php - description of the Russian translation	
redirect_and_exit()	165
redirect to another url by sending an http header	

S

sitemap_uninstall()	1208
uninstall the module	
sitemap_install()	1208
install the module	
sitemap_upgrade()	1209

<i>upgrade the module</i>	1210
sitemap_manifest.php	1210
<i>/program/modules/sitemap/sitemap_manifest.php - description of the sitemap module</i>	
sitemap_search()	1211
<i>search this module's content in selected nodes for keywords in \$qwords</i>	
sitemap_search.php	1211
<i>/program/modules/sitemap/sitemap_search.php - interface to the search-part of this module</i>	
sitemap_demodata()	1207
<i>add demonstration data to the system</i>	
sitemap_install.php	1207
<i>/program/modules/sitemap/sitemap_install.php - installer of the sitemap module</i>	
sitemap_save()	1203
<i>save the modified content data of this module linked to node \$node_id</i>	
sitemap_get_dialogdef()	1203
<i>construct a dialog definition for the sitemap 'scope' value</i>	
sitemap_show_edit()	1204
<i>present the user with a dialog to modify the sitemap that is connected to node \$node_id</i>	
sitemap_cron.php	1206
<i>/program/modules/sitemap/sitemap_cron.php - interface to the cron-part of the sitemap module</i>	
sitemap_cron()	1206
<i>routine that is called periodically by cron</i>	
sitemap_view.php	1212
<i>/program/modules/sitemap/sitemap_view.php - interface to the view-part of the sitemap module</i>	
sitemap_tree_walk()	1212
<i>walk the tree and send to output in the form of nested unnumbered lists (uses recursion)</i>	
snapshots_get_dialogdef()	1220
<i>construct a dialog definition for the snapshots configuration data</i>	
snapshots_disconnect()	1220
<i>disconnect this module from a node</i>	
snapshots_save()	1220
<i>save the modified content data of this module linked to node \$node_id</i>	
snapshots_show_edit()	1221
<i>present the user with a dialog to modify the snapshots that are connected to node \$node_id</i>	
snapshots_cron()	1223
<i>routine that is called periodically by cron</i>	
snapshots_cron.php	1223
<i>/program/modules/snapshots/snapshots_cron.php - interface to the cron-part of the snapshots module</i>	
snapshots_connect()	1219
<i>connect this module to a node</i>	
snapshots_check_path()	1218
<i>validate and massage the user-supplied data path</i>	
snapshots_tabledefs.php	1215
<i>/program/modules/snapshots/install/snapshots_tabledefs.php - data definition for module</i>	
sitemap_view()	1213
<i>display the content of the sitemap linked to node \$node_id</i>	
snapshots.php	1216
<i>/program/modules/snapshots/languages/en/snapshots.php - translated messages for module (English)</i>	
snapshots.php	1217
<i>/program/modules/snapshots/languages/nl/snapshots.php - translated messages for module (Dutch)</i>	

snapshots_admin.php	1218
<i>/program/modules/snapshots/snapshots_admin.php - management interface for snapshots-module</i>	
sitemap_disconnect()	1203
<i>disconnect this module from a node</i>	
sitemap_connect()	1202
<i>connect this module to a node</i>	
search_construct_nodeslist()	1188
<i>construct a helper table with a list of node_id's that are to be searched</i>	
search_areas()	1188
<i>search through area titles</i>	
search_context()	1190
<i>return a ready-to-use context with highlights</i>	
search_get_places()	1191
<i>construct an ordered list of places to search perhaps with hints about # of hits</i>	
search_highlight()	1192
<i>highlight words in a lowercase search context snippet</i>	
search_get_qwords()	1191
<i>construct an array with keywords to look for</i>	
search()	1187
<i>perform an actual search</i>	
search_view.php	1187
<i>/program/modules/search/search_view.php - interface to the view-part of the search module</i>	
search_upgrade()	1183
<i>upgrade the module</i>	
search_uninstall()	1183
<i>uninstall the module</i>	
search_manifest.php	1185
<i>/program/modules/search/search_manifest.php - description of the search module</i>	
search_search.php	1186
<i>/program/modules/search/search_search.php - interface to the search-part of this module</i>	
search_search()	1186
<i>search this module's content in selected nodes for keywords in \$qwords</i>	
search_module()	1192
<i>kickstarter for calling modules' search function (after loading the source files)</i>	
search_navbar()	1193
<i>construct a navigation bar with 'previous' 1 2 3 4 ... 'next'</i>	
sitemap_tabledefs.php	1199
<i>/program/modules/sitemap/install/sitemap_tabledefs.php - data definition for module</i>	
search_where()	1196
<i>construct a where-clause to look for keywords in 1 or more fields</i>	
sitemap.php	1200
<i>/program/modules/sitemap/languages/en/sitemap.php - translated messages for module (English)</i>	
sitemap.php	1201
<i>/program/modules/sitemap/languages/nl/sitemap.php - translated messages for module (Dutch)</i>	
sitemap_admin.php	1202
<i>/program/modules/sitemap/sitemap_admin.php - management interface for sitemap-module</i>	
search_view_get_dialogdef()	1196
<i>construct a dialog definition for the visitor's search form</i>	
search_view_get_config()	1196
<i>retrieve all configuration data for this search</i>	

search_show_form()	1194
<i>display the search form</i>	
search_nodes()	1193
<i>search through node titles and link_texts</i>	
search_simple()	1194
<i>perform a simple linear search in a module table</i>	
search_view()	1195
<i>display the content of the search linked to node \$node_id</i>	
search_view_dialog_validate()	1195
<i>validate the data entered by the visitor</i>	
snapshots_install.php	1224
<i>/program/modules/snapshots/snapshots_install.php - installer of the snapshots module</i>	
snapshots_demodata()	1224
<i>add demonstration data to the system</i>	
SnapshotViewerInline::noscript_add_inline_show()	1240
<i>construct the necessary code to show N static images in case JavaScript is OFF</i>	
SnapshotViewerInline::javascript_add_inline_show()	1240
<i>construct the necessary Javascript-code to do the inline slideshow configuration</i>	
SnapshotViewerInline::run()	1241
<i>read configuration parameters and actually generate the inline slide show</i>	
schoolyard.php	1306
<i>/program/themes/schoolyard/languages/en/schoolyard.php - translated messages for theme (English)</i>	
schoolyard.class.php	1308
<i>/program/themes/schoolyard/schoolyard.class.php - implements the Schoolyard Theme by David Prousch</i>	
schoolyard.php	1307
<i>/program/themes/schoolyard/languages/nl/schoolyard.php - translated messages for theme (Nederlands)</i>	
SnapshotViewerInline::\$inline_show_width	1239
SnapshotViewerInline::\$inline_show_visible_images	1239
SnapshotViewer::view_snapshot()	1238
<i>display a single full-size snapshot scaled to the specified dimension</i>	
SnapshotViewer::view_slideshow()	1238
<i>show the regular thumbnails overview and then pop-up a full-screen slideshow on top</i>	
SnapshotViewer::view_thumbnails()	1238
<i>display snapshots in the form of 0 or more clickable thumbnails</i>	
SnapshotViewerInline	1238
<i>this class implements methods to display snapshots</i>	
SnapshotViewerInline::\$inline_show_height	1239
schoolyard_install.php	1309
<i>/program/themes/schoolyard/schoolyard_install.php -- installer of the schoolyard theme</i>	
schoolyard_demodata()	1309
<i>add demonstration data to the system</i>	
sophia_demodata()	1319
<i>add demonstration data to the system</i>	
sophia_install.php	1319
<i>/program/themes/sophia/sophia_install.php -- installer for the sophia theme</i>	
sophia_install()	1320
<i>install the theme</i>	
sophia_uninstall()	1320
<i>uninstall the theme</i>	
sophia_manifest.php	1322

/program/themes/sophia/sophia_manifest.php	- description of the sophia theme	
sophia_upgrade()		1321
<i>upgrade the theme</i>		
sophia.php		1318
<i>/program/themes/sophia/languages/nl/sophia.php - translated messages for theme (Nederlands)</i>		
sophia.php		1317
<i>/program/themes/sophia/languages/en/sophia.php - translated messages for theme (English)</i>		
schoolyard_install()		1310
<i>install the theme</i>		
schoolyard_get_properties()		1310
<i>construct a list of default properties for this theme</i>		
schoolyard_uninstall()		1311
<i>uninstall the theme</i>		
schoolyard_upgrade()		1311
<i>upgrade the theme</i>		
schoolyard_manifest.php		1313
<i>/program/themes/schoolyard/schoolyard_manifest.php - description of the schoolyard theme</i>		
SnapshotViewer::view_raw()		1237
<i>output the raw image data as a tab-delimited file</i>		
SnapshotViewer::run()		1237
<i>task dispatcher</i>		
SnapshotViewer		1232
<i>this class implements methods to display snapshots</i>		
snapshots_view()		1230
<i>display the snapshots from the directory linked to node \$node_id</i>		
SnapshotViewer::\$area_id		1232
SnapshotViewer::\$default_showtime		1232
SnapshotViewer::\$domain		1233
SnapshotViewer::\$dimension		1232
snapshots_view.php		1230
<i>/program/modules/snapshots/snapshots_view.php - interface to the view-part of the snapshots module</i>		
snapshots_search()		1229
<i>search this module's content in selected nodes for keywords in \$qwords</i>		
snapshots_uninstall()		1226
<i>uninstall the module</i>		
snapshots_install()		1225
<i>install the module</i>		
snapshots_upgrade()		1226
<i>upgrade the module</i>		
snapshots_manifest.php		1228
<i>/program/modules/snapshots/snapshots_manifest.php - description of the snapshots module</i>		
snapshots_search.php		1229
<i>/program/modules/snapshots/snapshots_search.php - interface to the search-part of this module</i>		
SnapshotViewer::\$header		1233
SnapshotViewer::\$introduction		1233
SnapshotViewer::get_snapshots_configuration()		1236
<i>retrieve configuration data for this set of snapshots</i>		
SnapshotViewer::get_snapshots()		1235
<i>retrieve all image files (snapshots) from directory \$path</i>		
SnapshotViewer::images_array()		1236

construct an image configuration array	
SnapshotViewer::javascript_add_configuration()	1237
construct configuration data for javascript processing	
SnapshotViewer::javascript_include_once()	1237
include an external javascript file once	
SnapshotViewer::add_snapshot_navbar()	1235
add a navigation bar / tool bar for a snapshot	
SnapshotViewer::\$variant	1234
SnapshotViewer::\$node_id	1234
SnapshotViewer::\$module_record	1233
SnapshotViewer::\$snapshots	1234
SnapshotViewer::\$snapshots_path	1234
SnapshotViewer::\$theme	1234
search_install()	1183
install the module	
search_demodata()	1182
add demonstration data to the system	
snapshots.php	526
/program/modules/snapshots/languages/bg/snapshots.php	
sitemap.php	525
/program/modules/sitemap/languages/bg/sitemap.php	
schoolyard.php	531
/program/themes/schoolyard/languages/bg/schoolyard.php	
sophia.php	532
/program/themes/sophia/languages/bg/sophia.php	
schoolyard.php	545
/program/themes/schoolyard/languages/da/schoolyard.php	
sitemap.php	541
/program/modules/sitemap/languages/da/sitemap.php	
schoolyard.php	511
/program/themes/schoolyard/languages/ar/schoolyard.php	
sitemap.php	507
/program/modules/sitemap/languages/ar/sitemap.php	
send_file_from_datadir()	186
the designated file is sent to the visitor	
string2time()	167
convert a string representation of a date/time to a timestamp	
show_manual()	195
redirect the user to a specific place in the manual OR show helpful message about downloading the manual	
show_screen_choose_language()	195
show a screen to the visitor presenting a choice between various available translations of the manual	
show_screen_download()	196
show a screen to the visitor hinting at downloading a manual archive from download.websiteatschool.eu	
sitemap.php	558
/program/modules/sitemap/languages/de/sitemap.php	
snapshots.php	559
/program/modules/snapshots/languages/de/snapshots.php	
schoolyard.php	615
/program/themes/schoolyard/languages/fa/schoolyard.php	
sitemap.php	611

/program/modules/sitemap/languages/fa/sitemap.php	624
sitemap.php	624
/program/modules/sitemap/languages/fi/sitemap.php	628
schoolyard.php	628
/program/themes/schoolyard/languages/fi/schoolyard.php	647
snapshots.php	647
/program/modules/snapshots/languages/fr/snapshots.php	646
sitemap.php	646
/program/modules/sitemap/languages/fr/sitemap.php	602
sophia.php	602
/program/themes/sophia/languages/es/sophia.php	601
schoolyard.php	601
/program/themes/schoolyard/languages/es/schoolyard.php	565
sophia.php	565
/program/themes/sophia/languages/de/sophia.php	564
schoolyard.php	564
/program/themes/schoolyard/languages/de/schoolyard.php	579
sitemap.php	579
/program/modules/sitemap/languages/el/sitemap.php	583
schoolyard.php	583
/program/themes/schoolyard/languages/el/schoolyard.php	596
sitemap.php	596
/program/modules/sitemap/languages/es/sitemap.php	167
short_datim()	167
<i>construct an abbreviated date/time from a full date/time string</i>	
sanitise_filename()	166
<i>sanitise a string to make it acceptable as a filename/directoryname</i>	
SORTBY_FILE_ASC	71
SORTBY_DATE_DESC	71
SORTBY_FILE_DESC	71
SORTBY_NONE	71
SORTBY_SIZE_DESC	71
SORTBY_SIZE_ASC	71
SORTBY_DATE_ASC	71
show_benefactor_logo()	65
<i>output the logos of zero, one or more of the Website</i>	
show_configuration_intro()	30
<i>display an introductory text for the configuration manager + menu</i>	
show_accounts_menu()	22
<i>display the account manager menu</i>	
show_configuration_menu()	30
<i>display the configuration manager menu</i>	
SQL_FALSE	41
<i>this circumvents the sub-optimal implementation of booleans in MySQL</i>	
SQL_TRUE	41
<i>this circumvents the sub-optimal implementation of booleans in MySQL</i>	
show_login()	92
<i>show complete login dialog and exit</i>	
statisticslib.php	102
<i>/program/lib/statisticslib.php - statistics</i>	
sessions_show_overview()	117
<i>show an overview of currently existing sessions</i>	
sessions_show_node_locked()	117

<i>show locked node information from a single (combined) record</i>	
sessions show_session()	117
<i>show session information from a single (combined) record</i>	
show_tools_intro()	117
<i>display an introductory text for tools + menu</i>	
show_tools_menu()	118
<i>display the tools menu</i>	
sessions show_header()	116
<i>show the header of the HTML-table</i>	
sessions show_footer()	116
<i>close the HTML-table</i>	
statistics get_pages()	103
<i>construct an ordered list of pages to show in a statistics report</i>	
statistics embargo()	102
<i>check ancestors for embargo</i>	
statistics show_area_menu()	103
<i>show a menu of available areas</i>	
sessions delete()	115
<i>handle confirmation and actual delete of a session</i>	
sessions show()	115
<i>show a table with sessions</i>	
schoolyard.php	652
<i>/program/themes/schoolyard/languages/fr/schoolyard.php</i>	
sophia.php	653
<i>/program/themes/sophia/languages/fr/sophia.php</i>	
schoolyard.php	885
<i>/program/themes/schoolyard/languages/ur/schoolyard.php</i>	
snapshots.php	880
<i>/program/modules/snapshots/languages/ur/snapshots.php</i>	
sophia.php	886
<i>/program/themes/sophia/languages/ur/sophia.php</i>	
sitemap.php	901
<i>/program/modules/sitemap/languages/vi/sitemap.php</i>	
schoolyard.php	907
<i>/program/themes/schoolyard/languages/vi/schoolyard.php</i>	
snapshots.php	902
<i>/program/modules/snapshots/languages/vi/snapshots.php</i>	
sitemap.php	879
<i>/program/modules/sitemap/languages/ur/sitemap.php</i>	
sophia.php	858
<i>/program/themes/sophia/languages/sv/sophia.php</i>	
sv_manifest.php	844
<i>/program/languages/sv/sv_manifest.php - description of the Swedish translation</i>	
sockerr()	821
sitemap.php	851
<i>/program/modules/sitemap/languages/sv/sitemap.php</i>	
snapshots.php	852
<i>/program/modules/snapshots/languages/sv/snapshots.php</i>	
schoolyard.php	857
<i>/program/themes/schoolyard/languages/sv/schoolyard.php</i>	
sophia.php	908
<i>/program/themes/sophia/languages/vi/sophia.php</i>	
sitemap.php	917

/program/modules/sitemap/languages/zh/sitemap.php	
search_save()	1178
<i>save the modified content data of this module linked to node \$node_id</i>	
search_get_dialogdef()	1178
<i>construct a dialog definition for the search 'scope' and other parameters</i>	
search_show_edit()	1179
<i>present the user with a dialog to modify the search that is connected to node \$node_id</i>	
search_cron.php	1181
<i>/program/modules/search/search_cron.php - interface to the cron-part of the search module</i>	
search_install.php	1182
<i>/program/modules/search/search_install.php - installer of the search module</i>	
search_cron()	1181
<i>routine that is called periodically by cron</i>	
search_disconnect()	1177
<i>disconnect this module from a node</i>	
search_connect()	1177
<i>connect this module to a node</i>	
search_tabledefs.php	1174
<i>/program/modules/search/install/search_tabledefs.php - data definition for module</i>	
schoolyard.php	921
<i>/program/themes/schoolyard/languages/zh/schoolyard.php</i>	
search.php	1175
<i>/program/modules/search/languages/en/search.php - translated messages for module (English)</i>	
search.php	1176
<i>/program/modules/search/languages/nl/search.php - translated messages for module (Dutch)</i>	
search_admin.php	1177
<i>/program/modules/search/search_admin.php - management interface for search-module</i>	
spellchecker.php	816
sk_manifest.php	810
<i>/program/languages/sk/sk_manifest.php - description of the Slovak translation</i>	
snapshots.php	710
<i>/program/modules/snapshots/languages/it/snapshots.php</i>	
sitemap.php	709
<i>/program/modules/sitemap/languages/it/sitemap.php</i>	
schoolyard.php	715
<i>/program/themes/schoolyard/languages/it/schoolyard.php</i>	
sophia.php	716
<i>/program/themes/sophia/languages/it/sophia.php</i>	
snapshots.php	732
<i>/program/modules/snapshots/languages/pap/snapshots.php</i>	
sitemap.php	731
<i>/program/modules/sitemap/languages/pap/sitemap.php</i>	
schoolyard.php	696
<i>/program/themes/schoolyard/languages/hu/schoolyard.php</i>	
sitemap.php	692
<i>/program/modules/sitemap/languages/hu/sitemap.php</i>	
snapshots.php	669
<i>/program/modules/snapshots/languages/fry/snapshots.php</i>	
sitemap.php	668
<i>/program/modules/sitemap/languages/fry/sitemap.php</i>	
schoolyard.php	674
<i>/program/themes/schoolyard/languages/fry/schoolyard.php</i>	

sophia.php	675
<i>/program/themes/sophia/languages/fry/sophia.php</i>	
sophia.php	683
<i>/program/themes/sophia/languages/hi/sophia.php</i>	
schoolyard.php	737
<i>/program/themes/schoolyard/languages/pap/schoolyard.php</i>	
sophia.php	738
<i>/program/themes/sophia/languages/pap/sophia.php</i>	
snapshots.php	795
<i>/program/modules/snapshots/languages/ru/snapshots.php</i>	
sitemap.php	794
<i>/program/modules/sitemap/languages/ru/sitemap.php</i>	
schoolyard.php	800
<i>/program/themes/schoolyard/languages/ru/schoolyard.php</i>	
sophia.php	801
<i>/program/themes/sophia/languages/ru/sophia.php</i>	
sh_manifest.php	805
<i>/program/languages/sh/sh_manifest.php - description of the Serbo-Croatian translation</i>	
sitemap.php	778
<i>/program/modules/sitemap/languages/ro/sitemap.php</i>	
sophia.php	771
<i>/program/themes/sophia/languages/pt/sophia.php</i>	
schoolyard.php	751
<i>/program/themes/schoolyard/languages/pl/schoolyard.php</i>	
sitemap.php	747
<i>/program/modules/sitemap/languages/pl/sitemap.php</i>	
sitemap.php	764
<i>/program/modules/sitemap/languages/pt/sitemap.php</i>	
snapshots.php	765
<i>/program/modules/snapshots/languages/pt/snapshots.php</i>	
schoolyard.php	770
<i>/program/themes/schoolyard/languages/pt/schoolyard.php</i>	
show_accounts_intro()	22
<i>display an introductory text for the account manager + menu</i>	

T

TranslateTool::\$languages	410
TranslateTool::\$domains	410
TranslateTool::\$output	410
TranslateTool::\$show_parent_menu	411
TranslateTool::code_highlight()	411
<i>highlight code constructs in texts that are to be translated</i>	
TranslateTool::a_param()	411
<i>shorthand for the anchor parameters that lead to the translate tool</i>	
TranslateTool	410
<i>Methods to access properties of a language and modify translations</i>	
Theme::show_tree_walk()	409
<i>workhorse for constructing recursive menu (walk the tree) along the breadcrumb trail</i>	
Theme::node2anchor()	407
<i>construct an anchor from a node record</i>	
Theme::get_quicktop()	406

construct a list of quicklinks for top of page (if any)	408
Theme::queue_alert()	408
add \$message to alerts watching this page	
Theme::send_headers()	408
send collected HTTP-headers to user's browser	
Theme::set_preview_mode()	408
set the preview mode	
Theme::send_output()	408
send collected output to user's browser	
TranslateTool::diff_to_text()	412
convert an array with key-value-pairs to a php source file that can be included as a user translation	
TranslateTool::get_dialogdef language()	413
construct the language dialog (used for both add and edit)	
TranslateTool::language_save()	417
validate and save modified data to database	
TranslateTool::language_edit()	417
show the language edit dialog	
TranslateTool::language_savenew()	418
save the newly added language to the database	
TranslateTool::put_strings_userfile()	418
save new or changed translations to a file under CFG->datadir/languages	
TranslateTool::show_domain_menu()	419
display the domain menu via \$this->output	
TranslateTool::render_translation_dialog()	419
render a translation dialog based on a dialog definition	
TranslateTool::language_add()	417
present the language dialog where the user can enter properties for a new language	
TranslateTool::languages_overview()	416
display list of languages with edit icons and an option to add a language	
TranslateTool::get_domains()	414
return an ordered list of translation domains	
TranslateTool::get_dialogdef language domain()	413
construct the translation dialog for selected language and domain	
TranslateTool::get_icon_edit()	414
construct a clickable icon to edit the properties of this language	
TranslateTool::get_options_languages()	415
fetch a list of languages available as parent language	
TranslateTool::guess_row_count()	416
try to calculate a reasonable number of textarea rows based on the contents of \$text	
TranslateTool::get_strings_system()	415
retrieve strings (translations) and comments from an official (system) translation file	
Theme::get_quicklinks()	406
workhorse for constructing list of quicklinks	
Theme::get_quickbottom()	405
construct a list of quicklinks for bottom of page (if any)	
Theme::add_popup_bottom()	398
add a message to the list of popup-messages at the BOTTOM of the document	
Theme::add_meta_http_equiv()	398
add a line with http-equiv meta-information to the HTML head part of the document	
Theme::add_popup_top()	398
add a message to the list of popup-messages at the TOP of the document	
Theme::add_stylesheetsheet()	398

<i>add a link to a stylesheet to the HTML head part of the document</i>	
Theme::construct_tree()	399
<i>read all nodes from table for this area and construct a tree</i>	
Theme::calc_breadcrumb_trail()	399
<i>set breadcrumbs in tree AND construct list of clickable anchors</i>	
Theme::add_meta()	397
<i>add a line with meta-information to the HTML head part of the document</i>	
Theme::add_message()	397
<i>add a message to the list of inline messages, part of the BODY of the document</i>	
Theme::\$title	396
Theme::\$theme_record	396
Theme::\$tree	396
Theme::add_content()	397
<i>add a line or array of lines to the content part of the document</i>	
Theme::add_http_header()	397
<i>add an HTTP-header</i>	
Theme::add_html_header()	397
<i>add a header to the HTML head part of the document</i>	
Theme::dump_subtree()	399
<i>a helper-routine during development/debugging (currently unused)</i>	
Theme::get_address()	400
<i>return the reconstructed URL in a single (indented) line</i>	
Theme::get_logo()	403
<i>construct an image tag with the area logo</i>	
Theme::get_lines()	403
<i>get lines from an array in a single properly indented string</i>	
Theme::get_menu()	404
<i>construct the submenu starting at \$menu_id OR the first breadcrumb in the top level menu</i>	
Theme::get_navigation()	404
<i>construct a top level menu (navigation bar) as an unnumbered list (UL) of list items (LI)</i>	
Theme::get_properties()	405
<i>retrieve configuration parameters for this combination of theme and area</i>	
Theme::get_popups()	405
<i>construct javascript alerts for messages</i>	
Theme::get_jumpmenu()	402
<i>construct a simple jumplist to navigate to other areas</i>	
Theme::get_html_head()	402
<i>get all lines in the HTML head section in a single, properly indented string</i>	
Theme::get_bottomline()	400
<i>show 'powered by' and (maybe) report basic performance indicators</i>	
Theme::get_bazaar_style_style()	400
<i>collect bazaar style style from area and nodes</i>	
Theme::get_content()	401
<i>get all lines in the content DIV in a single properly indented string</i>	
Theme::get_div_breadcrumbs()	401
<i>construct breadcrumb trail</i>	
Theme::get_html()	402
<i>construct an output page in HTML</i>	
Theme::get_div_messages()	401
<i>get a perhaps bulleted list of messages in a DIV</i>	
TranslateTool::show_parent_menu()	420
<i>allow the caller to use the menu area (or not)</i>	
TranslateTool::submit_diff_to_project()	420

send new or changed translations back to the project	
ThemeRuta::get_html()	1300
ThemeRuta::get_bottomline()	1300
show footer text, maybe some quicklinks and 'powered by'	
ThemeRuta::get_menu()	1300
construct the submenu starting at \$menu_id OR the first breadcrumb in the top level menu	
ThemeRuta::get menunavigation()	1301
construct a top level menu (navigation bar)	
ThemeRuta::ruta_get_background()	1302
compute background image(s) based on current time and index	
ThemeRuta::get_navigation()	1301
construct a top level menu (navigation bar) with an additional menu button	
ThemeRuta::get bazaar_style_style()	1300
insert special rotating background banner in bazaar style style in header	
ThemeRuta::get_altnavigation()	1299
create alternative navigation in case Javascript is not available	
ThemeRosalina::rosalina_navigation_menu()	1289
construct the navigation menu	
ThemeRosalina::rosalina_menuwidth()	1289
calculate the maximum-width of the items in the section (menu) starting at \$node_id	
ThemeRosalina::rosalina_show_tree_walk()	1289
this treewalker shows the current menu and descends recursively	
ThemeRuta	1298
/program/themes/ruta/ruta.class.php - the class that implements the theme	
ThemeRuta::alt_show_tree_walk()	1299
workhorse for constructing alternative recursive menu (walk the tree)	
ThemeRuta::\$ruta_header_background	1298
ThemeRuta::ruta_get_sidebar()	1303
retrieve data for sidebar	
ThemeRuta::ruta_get_sidebar_htmlpage()	1303
retrieve page data for htmlpage module on a node	
ThemeSophia::get_bottomline()	1323
show footer text, maybe some quicklinks and 'powered by'	
ThemeSophia	1323
/program/themes/sophia/sophia.class.php - the class that implements the theme	
ThemeSophia::get_html()	1324
construct an output page in HTML	
ThemeSophia::get_menu()	1324
construct the submenu starting at \$menu_id OR the first breadcrumb in the top level menu	
TODO	1337
ThemeSophia::get_navigation()	1325
construct a top level menu (navigation bar) as an unnumbered list (UL) of list items (LI)	
ThemeSchoolyard::schoolyard_printpage()	1315
construct a 'print this page' link	
ThemeSchoolyard::schoolyard_logout()	1315
conditionally construct a logout link	
ThemeRuta::ruta_sidebar_nodes_modules()	1304
compute the list of sidebar blocks for a main menu item	
ThemeRuta::ruta_navigation_index()	1303
calculate the index of the current main navigation item	
ThemeRuta::show_menu_js()	1304
insert code for toggling the menu/free HTML in the left hand margin	
ThemeSchoolyard	1314

<i>this class implements the schoolyard theme (based on a design by David Prousch)</i>	
ThemeSchoolyard::schoolyard_get_div_quicktop()	1315
<i>construct an optional div for quicklinks at the top if any</i>	
ThemeSchoolyard::get_html()	1314
<i>construct an output page in HTML</i>	
ThemeRosalina::rosalina_menucount()	1288
<i>calculate the number of items in the section (menu) starting at \$node_id</i>	
ThemeRosalina::rosalina_hvmenu_config()	1288
<i>construct the necessary JavaScript that HV Menu needs</i>	
ThemeCornelia	1261
<i>/program/themes/cornelia/cornelia.class.php - the class that implements the theme</i>	
ThemeAxis::get_html()	1252
<i>construct an output page in HTML</i>	
ThemeCornelia::cornelia_get_background_url()	1261
<i>compute URL for a background image based on current time and index</i>	
ThemeCornelia::cornelia_get_sidebar()	1262
<i>retrieve data for sidebar</i>	
ThemeCornelia::cornelia_navigation_index()	1263
<i>calculate the index of the current main navigation item</i>	
ThemeCornelia::cornelia_get_sidebar_htmlpage()	1263
<i>retrieve page data for htmlpage module on a node</i>	
ThemeAxis::axis_printpage()	1251
<i>construct a 'print this page' link</i>	
ThemeAxis::axis_logout()	1251
<i>conditionally construct a logout link</i>	
TranslateTool::translation_save()	420
<i>save the modified translations in a file in the tree CFG->datadir/languages/</i>	
TranslateTool::translation_edit()	420
<i>show an edit dialog with phrases from \$full_domain in \$language_key</i>	
tabledata.php	467
<i>/program/install/tabledata.php defines core data in a generic way</i>	
tabledefs.php	468
<i>/program/install/tabledefs.php defines all core tables in a generic way</i>	
ThemeAxis	1251
<i>this class implements the axis theme</i>	
tr_manifest.php	863
<i>/program/languages/tr/tr_manifest.php - description of the Turkish translation</i>	
ThemeCornelia::cornelia_sidebar_nodes_modules()	1263
<i>compute the list of sidebar blocks for a main menu item</i>	
ThemeCornelia::get_bottomline()	1264
<i>show footer text, maybe some quicklinks and 'powered by'</i>	
ThemeRosalina::get_menu_areas()	1286
<i>construct a simple UL-based jump menu to select another area (when no Javascript is available)</i>	
ThemeRosalina::get_logo()	1285
<i>construct an image tag with the area logo</i>	
ThemeRosalina::get_quickbottom()	1286
<i>construct a list of quicklinks for bottom of page (if any) ending with a separator</i>	
ThemeRosalina::rosalina_get_page_head()	1286
<i>construct the page top</i>	
ThemeRosalina::rosalina_hvmenu()	1288
<i>construct the necessary JavaScript code for definition of HV Menu</i>	
ThemeRosalina::rosalina_hotspot_map()	1287

create a hotspot map for the logo	
ThemeRosalina::get_html()	1285
construct an output page in HTML	
ThemeRosalina::\$menu_top	1284
ThemeCornelia::get_menu()	1265
construct the submenu starting at \$menu_id OR the first breadcrumb in the top level menu	
ThemeCornelia::get_html()	1264
construct an output page in HTML	
ThemeCornelia::get_quicktop()	1265
construct a list of quicklinks for top of page (if any) + (maybe) a print-button	
ThemeFrugal	1274
/program/themes/frugal/frugal.class.php - the class that comprises the most minimal theme	
ThemeRosalina::\$menu_sub	1284
ThemeRosalina	1284
this class implements the rosalina theme (based on HV Menu)	
Theme::\$theme_id	395
Theme::\$text_only	395
TASK_MODULEMANAGER_EDIT	97
TASK_REMOVE_MULTIPLE_FILES	71
TASK_MODULEMANAGER_INTRO	97
TASK_MODULEMANAGER_SAVE	97
TASK_ADD_SECTION	100
TASK_ADD_PAGE	100
TASK_REMOVE_FILE	71
TASK_REMOVE_DIRECTORY	71
TASK_ADD_DIRECTORY	71
TASK_SITE	29
TASK_ADD_FILE	71
TASK_CHANGE_DIRECTORY	71
TASK_PREVIEW_FILE	71
TASK_LIST_DIRECTORY	71
TASK_NODE_DELETE	100
TASK_NODE_EDIT	100
TASK_SUBTREE_COLLAPSE	101
TASK_SET_DEFAULT	101
TASK_SUBTREE_EXPAND	101
TASK_TREEVIEW	101
TREE_VIEW_CUSTOM	101
TASK_TREEVIEW_SET	101
TASK_SAVE_NODE	101
TASK_SAVE_NEWSECTION	101
TASK_NODE_EDIT_CONTENT	101
TASK_NODE_EDIT_ADVANCED	101
TASK_NODE_FORCE_UNLOCK	101
TASK_PAGE_PREVIEW	101
TASK_SAVE_NEWPAGE	101
TASK_SAVE_CONTENT	101
TASK_CONFIGURATION_INTRO	29
TASK_AREAS	29
TASK_GROUP_DELETE	21
TASK_GROUP_CAPACITY_SAVE	21
TASK_GROUP_EDIT	21
TASK_GROUP_SAVE	21

TASK_USERS	21
<i>TASK_USER* relate to user accounts</i>	
TASK_GROUP_SAVE_NEW	21
TASK_GROUP_CAPACITY_PAGEMANAGER	21
TASK_GROUP_CAPACITY_OVERVIEW	21
<i>TASK_GROUP_CAPACITY_* relate to group-capacity-combinations</i>	
TASK_GROUPS	21
<i>TASK_GROUP* relate to plain groups</i>	
TASK_ACCOUNTS	21
<i>default selection for account manager: show introduction + links to users and groups</i>	
TASK_GROUP_ADD	21
TASK_GROUP_CAPACITY_ADMIN	21
TASK_GROUP_CAPACITY_MODULE	21
TASK_GROUP_CAPACITY_INTRANET	21
TASK_USER_ADD	22
TASK_USER_ADMIN	22
TASK_USER_PAGEMANAGER	22
TASK_USER_MODULE	22
TASK_USER_SAVE	22
TASK_USER_SAVE_NEW	22
TASK_ALERTS	29
TASK_USER_TREEVIEW	22
TASK_USER_INTRANET	22
TASK_USER_GROUPS_SAVE	22
TASK_USER_DELETE	22
TASK_USER_ADVANCED	22
TASK_USER_EDIT	22
TASK_USER_GROUP_ADD	22
TASK_USER_GROUPS	22
TASK_USER_GROUP_DELETE	22
TREE_VIEW_MAXIMAL	101
TREE_VIEW_MINIMAL	101
Theme	391
<i>Methods to access properties of a theme</i>	
tree_visibility()	169
<i>calculate the visibility of the nodes in the tree</i>	
Theme::\$area_id	391
Theme::\$area_record	391
Theme::\$breadcrumb_separator	392
Theme::\$breadcrumb_addendum	391
tree_build()	168
<i>construct a tree of nodes in memory</i>	
t()	167
<i>translation of phrases via a function with a very short name</i>	
TASK_UPDATE_CORE	121
TASK_INSTALL_THEME	121
TASK_UPDATE_LANGUAGE	122
TASK_UPDATE_MODULE	122
TASK_UPDATE_THEME	122
TASK_UPDATE_OVERVIEW	122
Theme::\$config	392
Theme::\$content	392
Theme::\$node_record	394

Theme::\$node_id	394
Theme::\$preview_mode	394
Theme::\$quickbottom_separator	395
Theme::\$silent_mode	395
Theme::\$quicktop_separator	395
Theme::\$messages_top	394
Theme::\$messages_inline	394
Theme::\$dtd	392
Theme::\$domain	392
Theme::\$html_head	393
Theme::\$http_headers	393
Theme::\$messages_bottom	393
Theme::\$jumps	393
TASK INSTALL MODULE	121
TASK INSTALL LANGUAGE	121
token_store()	111
<i>write the (serialised) data to the database</i>	
token_lookup()	111
<i>lookup \$reference + \$token_key in the table and retrieve token information</i>	
toolslib.php	113
<i>/program/lib/toolslib.php - tools</i>	
TASK BACKUPTOOL	113
TASK SESSIONTOOL	113
TASK LOGVIEW	113
token_garbage_collect()	110
<i>remove all expired tokens</i>	
token_fetch()	110
<i>retrieve the (unserialised) data from the database</i>	
themelib.php	106
<i>/program/lib/themelib.php - theme factory</i>	
theme.class.php	105
<i>/program/lib/theme.class.php - taking care of themes</i>	
theme_factory()	106
<i>manufacture a theme object</i>	
tokenlib.php	108
<i>/program/lib/tokenlib.php - functions to manipulate unique tokens via the database</i>	
token_destroy()	110
<i>remove a token record from the tokens table (it should still exist)</i>	
token_create()	108
<i>create a new record in the tokens table, return the unique token_id</i>	
TASK TOOLS INTRO	113
TASK TRANSLATETOOL	113
TRANSLATETOOL CHORE LANGUAGE SAVE NEW	120
TRANSLATETOOL CHORE LANGUAGE SAVE	120
TRANSLATETOOL CHORE OVERVIEW	120
TRANSLATETOOL CHORE SAVE	120
TRANSLATETOOL PARAM LANGUAGE KEY	120
<i>This parameter identifies the language.</i>	
TRANSLATETOOL PARAM DOMAIN	120
TRANSLATETOOL CHORE LANGUAGE EDIT	120
TRANSLATETOOL CHORE LANGUAGE ADD	120
task_logview()	119
<i>quick and dirty logfile view / download / prune tool</i>	

task_backuptool()	118
<i>show an introductory text for backup tool OR stream a ZIP-file to the browser</i>	
task_sessiontool()	119
<i>main entry point for session management</i>	
tools_get_hostname()	119
<i>helper routine to construct something that looks like this hosts name</i>	
TRANSLATETOOL_CHORE_EDIT	120
translatetool.class.php	120
<i>/program/lib/translatetool.class.php - taking care of language translations</i>	
THUMBNAIL_PREFIX	10
<i>This global constant is used to specify thumbnail files to be ignored in directory listings</i>	

U

UserManager::areas_expand_collapse()	432
<i>manipulate the current state if indicator(s) for 'open' and 'closed' areas</i>	
UserManager::a_params()	432
<i>shorthand for the anchor parameters that lead to the user manager</i>	
UserManager::\$users	432
UserManager::\$show_parent_menu	431
UserManager	431
<i>User management</i>	
UserManager::\$output	431
UserManager::calc_acl_id()	433
<i>determine the acl_id for user user_id</i>	
UserManager::delete_user_records()	433
<i>remove all records relating to a single acl_id from various acl-tables</i>	
UserManager::get_fname()	434
<i>shorthand for the first readable name in a dialogdef item</i>	
UserManager::get_icon_delete()	434
<i>construct a clickable icon to delete this user</i>	
UserManager::get_dialogdef_edit_user()	434
<i>construct the edit user dialog</i>	
UserManager::get_dialogdef_confirm_delete()	434
<i>construct the edit user dialog</i>	
UserManager::get_dialogdef_add_user()	433
<i>construct the add userdialog</i>	
UserManager::get_dialogdef_add_usergroup()	433
<i>construct a dialogdef for selecting a group/capacity</i>	
Useraccount::where_acl_id()	431
<i>a convenient routine to construct a selection of acls</i>	
Useraccount::is_admin_pagemanager()	430
<i>determine whether the user has administrator privilege for pagemanager</i>	
Useraccount::has_area_permissions()	427
<i>determine user's permissions for an area</i>	
Useraccount::has_intranet_permissions()	428
<i>determine user's permissions for an intranet area</i>	
Useraccount::fetch_acls_from_table()	427
<i>retrieve acl-data from table into a sparse array</i>	
Useraccount::\$user_id	426
Useraccount::\$skin	426
Useraccount::\$username	426

Useraccount::has_job_permissions()	428
determine user's permissions for a job	
Useraccount::has_module_area_permissions()	428
determine user's permissions for a module at the area level	
Useraccount::has_site_permissions()	430
determine user's permissions for the site-level	
Useraccount::is_admin()	430
determine whether the user has administrator privilege	
Useraccount::has_node_permissions()	429
determine user's permissions for a node within an area	
Useraccount::has_module_site_permissions()	429
determine user's permissions for a module at the site-level	
Useraccount::has_module_node_permissions()	428
determine user's permissions for a module at the node level	
UserManager::get_icon_edit()	435
construct a clickable icon to edit the properties of this user	
UserManager::get_icon_groupdelete()	435
construct a clickable icon to delete a membership from this user	
UserManager::user_groups()	443
present an overview of group memberships for the specified user	
UserManager::user_groupsave()	443
save the new group/capacity for the selected user	
UserManager::user_groupdelete()	442
end the group membership for the selected user	
UserManager::user_groupadd()	442
present 'add membership' dialog	
UserManager::user_delete()	441
delete a user after confirmation	
UserManager::user_edit()	442
present an 'edit user' dialog filled with existing data	
UserManager::user_intranet()	443
show a dialog for modifying intranet permissions for a user	
UserManager::user_pagemanager()	443
show a dialog for modifying page manager permissions for a user	
utf16_strlen()	821
ur_manifest.php	872
/program/languages/ur/ur_manifest.php - description of the Urdu translation	
UserManager::user_save_basic()	444
save basic properties of user account	
UserManager::user_savenew()	444
save a new user to the database	
UserManager::user_save()	444
save edited user data to the database	
UserManager::user_admin()	441
show a dialog for modifying admin permissions for a user	
UserManager::user_add()	440
present 'add user' dialog where the user can enter minimal properties for a new user	
UserManager::get_user_records()	437
retrieve (a selection of) all user records from the database	
UserManager::has_job_permission()	437
determine whether a user has permissions for a particular job	
UserManager::get_user_record()	437
retrieve a single user's record possibly from the cache	

<u>UserManager::get_user_names()</u>	436
<i>shortcut to retrieve the username and full name of the selected user</i>	
<u>UserManager::get_num_user_records()</u>	435
<i>calculate the total number of users in a specific group</i>	
<u>UserManager::get_options_available_groups_capacities()</u>	436
<i>construct a list of groups still available for this user</i>	
<u>UserManager::show_breadcrumbs_adduser()</u>	438
<i>display breadcrumb trail that leads to the add new user dialog</i>	
<u>UserManager::show_breadcrumbs_overview()</u>	438
<i>display breadcrumb trail that leads to users overview screen</i>	
<u>UserManager::show_parent_menu()</u>	440
<u>UserManager::users_overview()</u>	440
<i>display a list of existing users and an option to add a user</i>	
<u>UserManager::show_menu_user()</u>	439
<i>show the user menu with current option highlighted</i>	
<u>UserManager::show_menu_overview()</u>	439
<i>display a menu showing groups of users (if any) + corresponding breadcrumb trail</i>	
<u>UserManager::show_breadcrumbs_user()</u>	438
<i>display breadcrumb trail that leads to the edit user dialog</i>	
<u>Useraccount::\$related_acls</u>	426
<u>Useraccount::\$properties</u>	426
<u>update_language()</u>	131
<i>update a language in the database</i>	
<u>update_module()</u>	132
<i>call the module-specific upgrade routine</i>	
<u>update_create_tables()</u>	131
<i>create tables in database via include()'ing a file with tabledefs</i>	
<u>update_create_table()</u>	131
<i>create table in database from an individual tabledef</i>	
<u>update_core_2016062900()</u>	130
<i>perform actual update to version 2016062900</i>	
<u>update_core_version()</u>	130
<i>record the specified version number in the config table AND in \$CFG->version</i>	
<u>update_remove_obsolete_files()</u>	132
<i>attempt to remove or at least flag obsolete files</i>	
<u>update_show_overview()</u>	133
<i>display an introductory text for update + status overview</i>	
<u>update_theme()</u>	134
<i>call the theme-specific upgrade routine</i>	
<u>useraccount.class.php</u>	135
<i>/program/lib/useraccount.class.php - taking care of useraccounts</i>	
<u>update_status_table_open()</u>	133
<i>open a status overview HTML-table including column headers</i>	
<u>update_status_table_close()</u>	133
<i>close the status overview HTML-table we opened before</i>	
<u>update_status_anchor()</u>	133
<i>return an anchor tag with link to the specific update function</i>	
<u>update_core_2014111700()</u>	130
<i>perform actual update to version 2014111700</i>	
<u>update_core_2013071100()</u>	130
<i>perform actual update to version 2013071100</i>	
<u>USERMANAGER_DIALOG_PAGEMANAGER</u>	22
<u>updatelib.php</u>	121

/program/lib/updatelib.php - update wizard	
USERMANAGER DIALOG INTRANET	22
USERMANAGER DIALOG EDIT	22
USERMANAGER DIALOG ADMIN	22
USERMANAGER DIALOG DELETE	22
update_core()	125
<i>update the core version in the database to the version in the version.php file (the 'manifest' version)</i>	
update_core_2010120800()	125
<i>perform actual update to version 2010120800</i>	
update_core_2011093000()	126
<i>perform actual update to version 2011093000</i>	
update_core_2012041900()	129
<i>perform actual update to version 2012041900</i>	
update_core_2011051100()	125
<i>perform actual update to version 2011051100</i>	
update_core_2011020100()	125
<i>perform actual update to version 2011020100</i>	
update_core_2010122100()	125
<i>perform actual update to version 2010122100</i>	
usermanager.class.php	138
<i>/program/lib/usermanager.class.php - taking care of user management</i>	
utf8lib.php	139
<i>/program/lib/utf8lib.php - utility-routines for UTF-8</i>	
Useraccount::\$scls_nodes	424
Useraccount::\$scl_id	424
Useraccount::\$scls_modules_nodes	424
Useraccount::\$scls_modules_areas	423
Useraccount::\$scls_areas	423
Useraccount::\$scls_modules	423
Useraccount::\$sarea_permissions_from_nodes	424
Useraccount::\$editor	424
Useraccount::\$language_key	425
Useraccount::\$path	425
Useraccount::\$is_logged_in	425
Useraccount::\$full_name	425
Useraccount::\$email	425
Useraccount::\$scls	423
Useraccount	421
<i>Methods to access properties of the account of the logged in user</i>	
utf8_strtolower()	141
<i>fold a UTF-8 string to lower case</i>	
utf8_strtoupper()	141
<i>fold a UTF-8 string to upper case (sort of)</i>	
utf8_strtoascii()	140
<i>map some UTF-8 characters to comparable ASCII strings</i>	
utf8_strlen()	140
<i>calculate the number of code points encoded in an UTF-8 string</i>	
USE_MBSTRING	140
utf8_strcasecmp()	140
<i>compare two UTF8 strings in a case-INsensitive way</i>	
utf8_substr()	142
<i>return part of a UTF-8 string</i>	

utf8_validate()	142
<i>check an arbitrary string for UTF-8 conformity</i>	
update_view_count()	193
<i>update the view count for page \$node_id</i>	
utf8lib.php	197
<i>/program/lib/utf8lib.php - utility-routines for UTF-8</i>	
update_statistics()	192
<i>update all statistics for the view of page \$node_id</i>	
userdir_is_empty()	170
<i>determine whether a directory is empty (free from (user)files)</i>	
userdir_delete()	170
<i>remove an 'empty' directory that used to contain (user)files</i>	
USERMANAGER_DIALOG_ADD	22

V

vi_manifest.php	892
<i>/program/languages/vi/vi_manifest.php - description of the Vietnamese translation</i>	
version.php	200
<i>'version.php' defines internal and external version numbers</i>	
valid_datetime()	63
<i>check validity of date, time or datetime</i>	

W

was.php	640
<i>/program/languages/fr/was.php</i>	
was.php	633
<i>/program/languages/fil/was.php</i>	
was.php	660
<i>/program/languages/fry/was.php</i>	
was.php	681
<i>/program/languages/hi/was.php</i>	
was.php	690
<i>/program/languages/hu/was.php</i>	
was.php	622
<i>/program/languages/fi/was.php</i>	
was.php	609
<i>/program/languages/fa/was.php</i>	
was.php	539
<i>/program/languages/da/was.php</i>	
was.php	552
<i>/program/languages/de/was.php</i>	
was.php	572
<i>/program/languages/el/was.php</i>	
was.php	592
<i>/program/languages/es/was.php</i>	
was.php	703
<i>/program/languages/it/was.php</i>	
was.php	725
<i>/program/languages/pap/was.php</i>	

was.php	864
was.php	845
was.php	873
was.php	893
was.php	914
was.php	811
was.php	806
was.php	745
was.php	758
was.php	776
was.php	788
was.php	518
was.php	505
WLOG_WARNING	11
<i>This global constant replaces a similar built-in constant LOG_WARNING which is erroneously defined as 5 in win32.h</i>	
WLOG_NOTICE	11
<i>This global constant replaces a similar built-in constant LOG_NOTICE which is erroneously defined as 6 in win32.h</i>	
wasentry script_name()	13
<i>determine the name of the executing script (the entry point)</i>	
was_version_check()	13
<i>check version of PHP-files against version stored in database</i>	
was.php	17
<i>/program/languages/en/was.php - generic translated messages</i>	
WLOG_INFO	11
<i>This global constant replaces a similar built-in constant LOG_INFO which is defined as 6 in win32.h</i>	
WLOG_ERR	11
<i>This global constant replaces a similar built-in constant LOG_ERR which is erroneously defined as 4 in win32.h</i>	
WLOG_ALERT	10
<i>This global constant replaces a similar built-in constant LOG_ALERT which is defined as 1 in win32.h</i>	
WLOG_CRIT	10
<i>This global constant replaces a similar built-in constant LOG_CRIT which is erroneously defined as 1 in win32.h</i>	
WLOG_DEBUG	11
<i>This global constant replaces a similar built-in constant LOG_DEBUG which is erroneously defined as 6 in win32.h</i>	

WLOG_EMERG	11
<i>This global constant replaces a similar built-in constant LOG_EMERG which is erroneously defined as 1 in win32.h</i>	
was.php	20
<i>/program/languages/nl/was.php - generic translated messages (Dutch)</i>	
was_login()	94
<i>execute the selected login procedure</i>	
WAS_RELEASE	201
<i>The external version number, like 1.0 or 1.0.0</i>	
WAS_ORIGINAL	201
<i>A boolean flag indicating this is either the original (TRUE) or a modified (FALSE) version of Website@School</i>	
WAS_RELEASE_DATE	201
<i>Date of distribution file generation in ISO 8601 format: yyyy-mm-dd OR yyyy-mm-ddThh:mm:ss+0000</i>	
WAS_VERSION	201
<i>The internal version number, like 2008012873 or 2008020100 (31 bits will work until the year 2147)</i>	
WASENTRY	456
<i>Valid entry points define WASENTRY; prevents direct access to include()'s.</i>	
was_url()	172
<i>massage a possibly relative URL to make it more qualified</i>	
was_node_url()	171
<i>construct a ready-to-use href which links to the node \$node via index.php</i>	
was_logout()	95
<i>end a session (logout the user) and maybe redirect</i>	
was_password_hash()	96
<i>calculate a hash from a salt and a password</i>	
waslib.php	144
<i>/program/lib/waslib.php - core functions</i>	
was_file_url()	170
<i>construct a url that links to a file via /file.php</i>	
WASENTRY	8
<i>Valid entry points define WASENTRY; prevents direct access to include()'s.</i>	

Z

Zip::dos_time_date()	449
<i>construct an MS-DOS time and date based on unix timestamp</i>	
Zip::make_suitable_filename()	450
<i>construct a suitable filename for use in ZIP-archive</i>	
Zip::CloseZip()	449
<i>finish the ZIP-archive by outputting the central directory and closing output</i>	
Zip::AddFile()	449
<i>add the contents of an existing file to the current ZIP-archive</i>	
Zip::\$zip_type	448
Zip::AddData()	449
<i>add data to the current ZIP-archive</i>	
Zip::OpenZipbuffer()	450
<i>prepare the user supplied buffer for subsequent ZIP-archive data</i>	
Zip::OpenZipfile()	451
<i>open a file for subsequent output of ZIP-archive</i>	

Zip::zip_error()	452
<i>add an error message to the list of error messages</i>	
zh_manifest.php	915
<i>/program/languages/zh/zh_manifest.php - description of the Chinese translation</i>	
Zip::zip_add_directories()	452
<i>workhorse function to add 0, 1 or more directories to the current ZIP-archive</i>	
Zip::zip_add_data()	451
<i>workhorse function to add data to the current ZIP-archive</i>	
Zip::OpenZipstream()	451
<i>start with a stream (direct output) indicating an application/zip type of content</i>	
Zip::\$zip_path	448
Zip::\$zip_filehandle	448
ZIP_TYPE_STREAM	174
zip.class.php	199
<i>/program/lib/zip.class.php - create simple ZIP-archives</i>	
ZIP_TYPE_NONE	174
ZIP_TYPE_FILE	174
ZIP_TYPE_BUFFER	174
Zip	445
<i>Create simple and compatible ZIP-archives</i>	
Zip::\$central_directory	447
Zip::\$zip_buffer	448
Zip::\$zip_comment	448
Zip::\$offset	447
Zip::\$no_name_files	447
Zip::\$Error	447
zip.class.php	174
<i>/program/lib/zip.class.php - create simple ZIP-archives</i>	