



EIEL-Utilidades

ESPECIFICACIÓN DEL SUBSISTEMA

Proyecto	EIEL-Utilidades
Responsable	David Trillo Pérez
Versión	0.2
Fecha	15/04/2008
Estado	Final
Clasificación	Público

Índice

1. Introducción.....	3
2. Arquitectura global del sistema	5
Historial de Cambios	5

1. Introducción

Nombre del proyecto: [EIEL-Utilidades](#)

EIEL-Utilidades contiene utilidades varias para el desarrollo de aplicaciones.

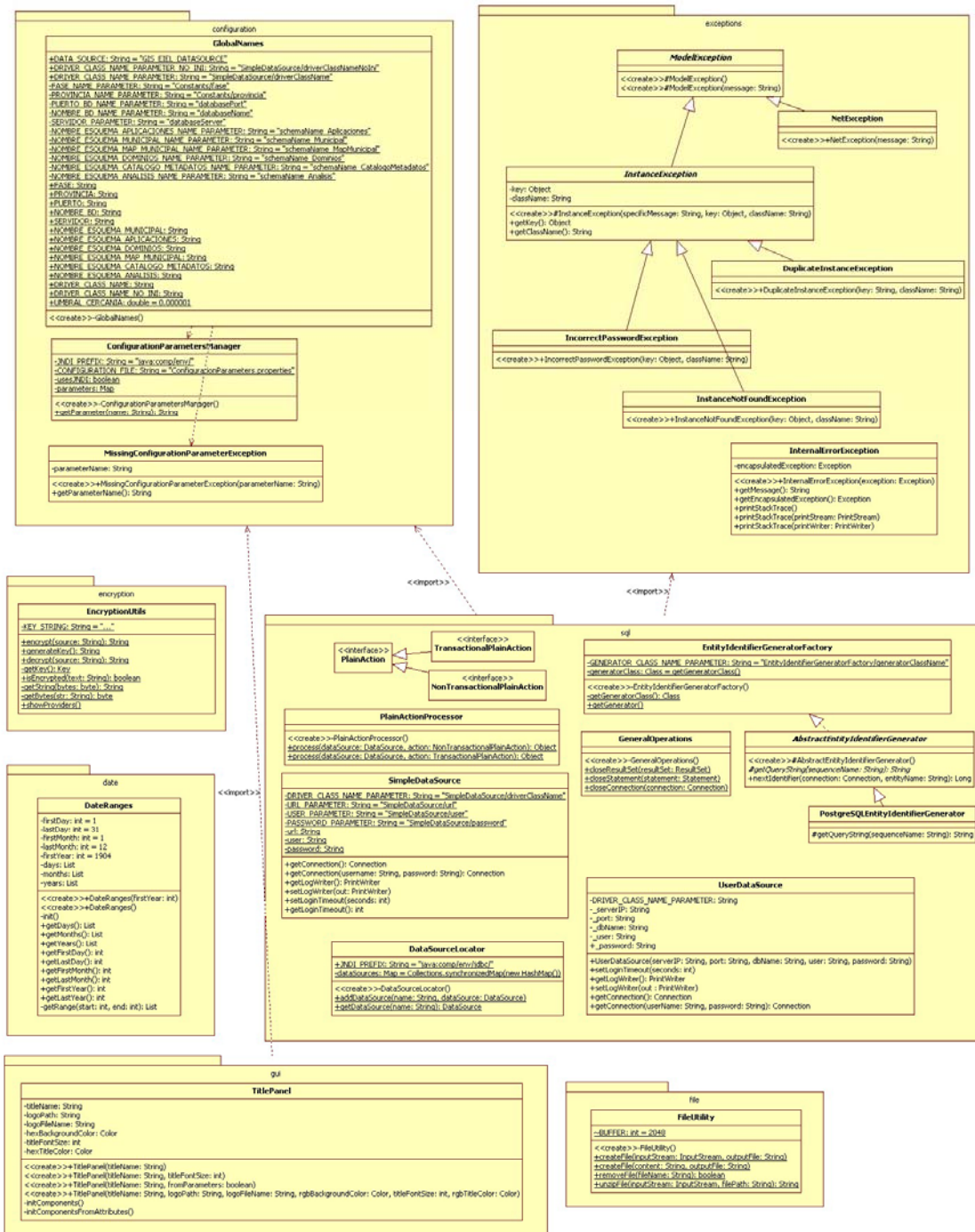
Estructura de directorios:

- bin: Contiene los .class del subsistema
- dist: Contiene EIEL-Utilidades.jar con las clases del subsistema.
- lib: Contiene las diferentes librerías utilizadas por el subsistema.
- src: Contiene los fuentes del subsistema.

Estructura de paquetes:

- Directorio “base” del subsistema: [es.udc.lbd.eiel.util](#).
- Estructura de directorios:
 - [es.udc.lbd.eiel.util.configuration](#): Utilidades para la gestión de parámetros de configuración y variables globales a través de ficheros de configuración.
 - `ConfigurationParametersManager`: Clase que permite obtener los valores de los parámetros de configuración a través de JNDI o de un fichero de configuración.
 - `MissingConfigurationParameterException`: Excepción que se lanzará cuando falte algún parámetro en el fichero de configuración.
 - `GlobalNames`: Clase estática que almacena las variables globales de la aplicación. Obtiene los valores de las variables a partir de la clase `ConfigurationParametersManager`.
 - [es.udc.lbd.eiel.util.date](#): Utilidades para la gestión de fechas.
 - `DateRanges`: Clase utilidad que permite obtener una lista de días, meses y años para mostrarlas en los formularios y para validar las fechas de los formularios.
 - [es.udc.lbd.eiel.util.encryption](#): Utilidades de cifrado simétrico.
 - `EncryptionUtils`: Clase utilidad que permite cifrar y descifrar texto con el algoritmo de cifrado simétrico DES, padding y ECB.
 - [es.udc.lbd.eiel.util.exceptions](#): Paquete con diferentes excepciones:
 - `DuplicateInstanceException`
 - `IncorrectPasswordException`
 - `InstanceException`
 - `InstanceNotFoundException`
 - `InternalErrorException`
 - `ModelException`
 - `NetException`
 - [es.udc.lbd.eiel.util.file](#): Paquete con diferentes utilidades para el manejo de ficheros de texto:
 - `FileUtility`: Clase utilidad que permite crear ficheros en disco que se suben desde las máquinas de los clientes, eliminar ficheros y descomprimirlos.

- [es.udc.lbd.eiel.util.gui](#): Paquete con elementos comunes utilizados en las interfaces de usuario:
 - `TitlePanel`: Panel de título genérico, utilizado en los formularios de la aplicación.
- [es.udc.lbd.eiel.util.sql](#): Utilidades sql:
 - `AbstractEntityIdentifierGenerator`: Implementación de la interfaz *EntityIdentifierGenerator*. Implementa el método *nextIdentifier(Connection, String)* requerido por el método *getQueryString(String)* que es utilizado por algunas clases.
 - `EntityIdentifierGenerator`: Define una interfaz para acceder al generador de identificadores para aquellos SGBD que lo soportan.
 - `EntityIdentifierGeneratorFactory`: Factoría para obtener objetos del tipo *EntityIdentifierGenerator*.
 - `PostgreSQLEntityIdentifierGeneratorFactory`: Implementación de *EntityIdentifierGenerator* para postgres.
 - `DataSourceLocator`: Esta clase cachea referencias a *DataSources*. Permite registrar un *DataSource* bajo un nombre y obtener una referencia a él usando ese nombre (si todavía no está registrado o cacheado usa JNDI para obtener la referencia).
 - `GeneralOperations`: Clase utilidad que evita duplicar código en todas las clases de acceso a datos ya que incluye métodos para llevar a cabo operaciones que se realizan en todas las clases de accesos a datos tales como cerrar *ResultSets*, cerrar *Connections*, etc.
 - `TransactionalPlainAction`: Una interface 'marker' para especificar acciones ordinarias transaccionales.
 - `NonTransactionalPlainAction`: Una interface 'marker' para especificar acciones no transaccionales.
 - `PlainAction`: Una interface base de todas las acciones 'ordinarias'.
 - `PlainActionProcessor`: Clase utilidad que permite ejecutar acciones 'ordinarias'. La ejecución de estas acciones será distinta según se trate de una *TransactionalAction* o de una *NonTransactionalAction*.
 - `SimpleDataSource`: Implementación de un *DataSource* que no hace pool de conexiones.
 - `UserDataSource`: Implementación de un *DataSource* para postgres.



Historial de Cambios

Versión	Fecha	Cambios
0.1	14/06/2007	David Trillo: Versión Inicial
0.2	15/04/2008	David Trillo: Iteración 3