**MEMORIA CIENTÍFICO TÉCNICA DEL PROYECTO**

**DESARROLLADORES: Carlos Robles, Jose Gracia**

**TÍTULO DEL PROYECTO:** ADMINISTRADOR DE INCIDENCIAS CENTRO EDUCATIVO

1. **RESUMEN DE LA PROPUESTA:**

**Objetivo:**

|  |
| --- |
| **Se precisa de un sistema para gestionar las incidencias de un centro y así llevar un seguimiento de las mismas. De esta forma se conseguirá un mejor rendimiento solucionando los problemas que surgen a lo largo del curso escolar.** |

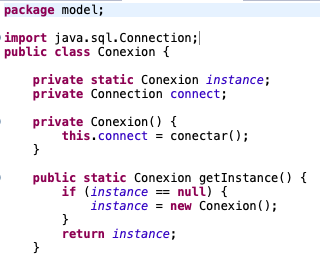
**Resumen:**

|  |
| --- |
| **La finalidad de esta aplicación es la gestión de las incidencias de un centro educativo. Con la misma se prodran crear, modificar y eliminar peticiones para solucionar problemas relacionados con los materiales del centro. Todas las incidencias se almacenarán en una base de datos, junto con las ubicaciones y usuarios que existen en el centro. Existirán 4 tipos de usuarios; profesor, jefe de departamento, mantenimiento, mensajes y administrador. Cada uno de los mismos trendran diferentes opciones y podrán intercambiarse mensajes con el fin de solucionar las incidencias que vayan surgiendo de una forma más dinámica.** |

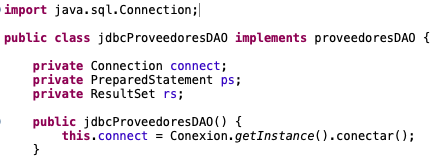
**2. Patrones:**

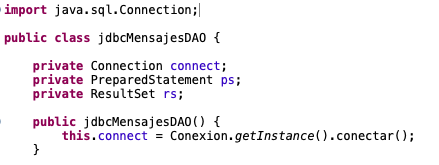
**Singleton**

* Diseño creado para restringir la creación de objetos pertenecientes a una clase.
* Garantiza que una clase solo tenga una instancia y proporciona un punto de acceso global a ella.
* Este patrón se ha aplicado de la siguiente forma:
  + - Se aplica en la clase raíz que se desea instanciar múltiples veces



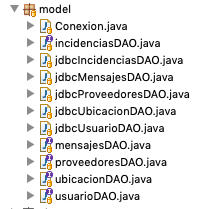
* + - Todas las clases que la instancien la llaman mediante *getInstance*





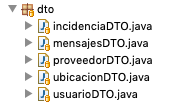
**DAO**

* El problema que vine a resolver este patrón es el acceso a los datos, que básicamente tiene que ver con la gestión de diversas fuentes de datas y además abstrae la forma de acceder a ellos
* Cada clase se sigue a su correspondiente interfaz.
* En nuestro caso este patrón se aplica en la implementacion de las clases de gestión de base de datos.
* Este patrón se ha aplicado de la siguiente forma:

****

**DTO**

* Crea objetos planos con una serie de atributos cuya utilidad solo puede ser de lectura; es decir han de evitar operaciones y métodos que realicen cálculos sobre los datos.

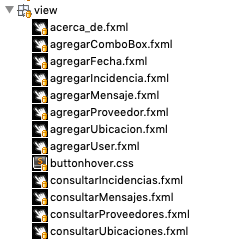
****

**Modelo vista controlador**

* En español Modelo Vista Controlador, este patrón permite separar una aplicación en 3 capas, una forma de organizar y de hacer escalable un proyecto, a continuación una breve descripción de cada capa.
* **Modelo**: Esta capa representa todo lo que tiene que ver con el acceso a datos: guardar, actualizar, obtener datos, además todo el código de la lógica del negocio, básicamente son las clases Java y parte de la lógica de negocio.



* **Vista**: La vista tiene que ver con la presentación de datos del modelo y lo que ve el usuario, por lo general una vista es la representación visual de un modelo.

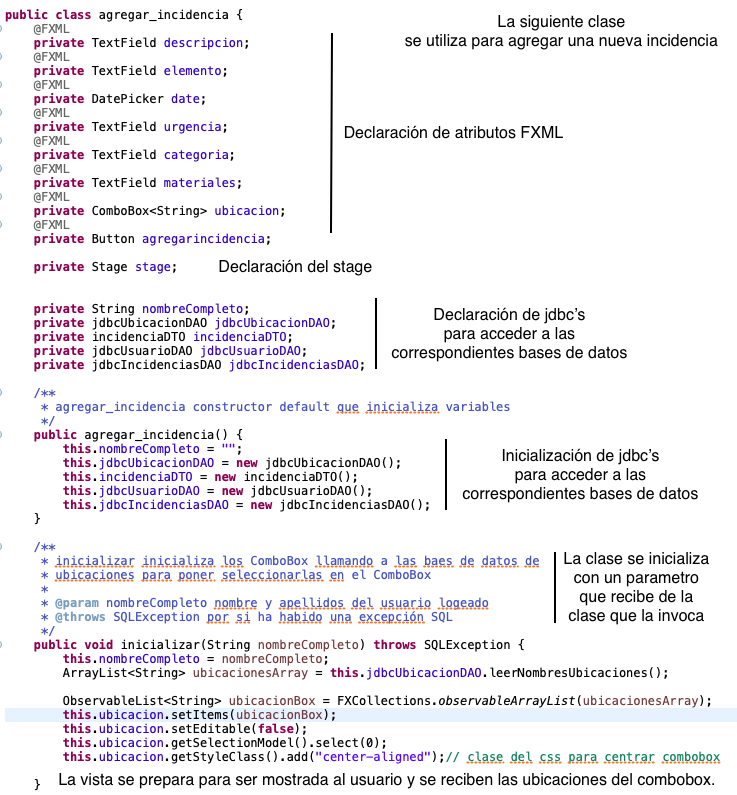


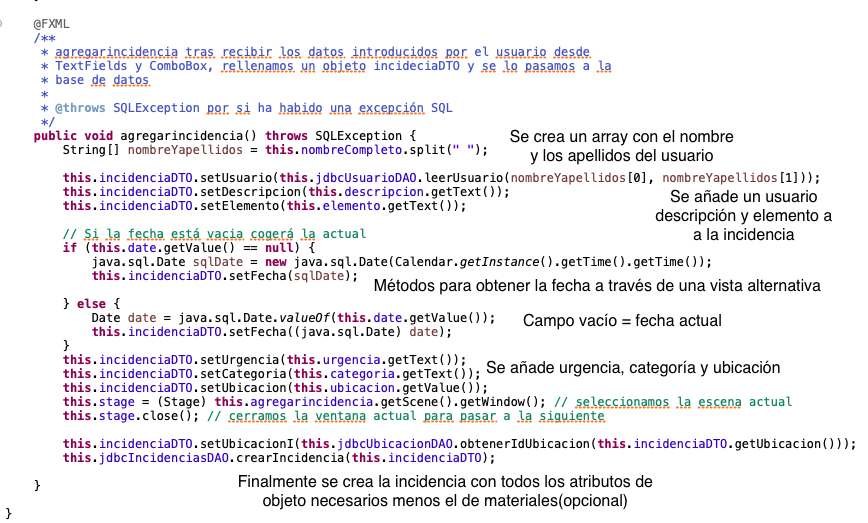
* **Controlador**: El controlador es el encargado de conectar el modelo con las vistas, funciona como un puente entre la vista y el modelo, el controlador recibe eventos generados por el usuario desde las vistas y se encargar de direccionar al modelo la petición respectiva.



**3. Explicar una clase: *agregar\_incidencia* de *controllerIncidencias***

* **Primera parte:**



****

**4. Problemas técnicos:**

**Estos son algunos de los problemas técnicos que hemos tenidos a lo largo de la creación de este proyecto:**

* **Paso de información entre interfaces.**
* **Acceso a la base de datos.**
* **Aplicación de los patrones DAO, DTO y Vista Controlador**
* **Cerrado de la base de datos.**
* **Distribución de clases.**
* **Filtrado de datos.**
* **Aplicación de instancias del patrón Singleton.**
* **Encriptación de credenciales.**
* **Distribución de paquetes.**
* **Creación del UML – Diagrama de clases**
* **Encriptaciónd de credenciales**

**5. Posibles mejoras:**

* **Importación de la base de datos a un hosting mysql online.**
* **Mejora de la base de datos mediante:**
  + - **Triggers**
    - **Eventos**
    - **Vistas**
* **Mejorar efectos visuales de las interfaces.**
* **Compactación de código.**

**6. Valoración personal:**

En este proyecto hemos mejorado notablemente nuestros conocimientos en java. Nos hemos encontrado ante bastantes errores pero hemos sabido solucionarlos desde un principio gracias a diferentes lugares de la web como stackoverflow. Hemos estado al día de las últimas novedades de javafx, las cuales han sido aplicadas satisfactoriamente al proyecto. Esperamos la aprobación de Borja.

Jose y Carlos.