

Desarrollo de Interfaces - 2º DAM

Manual Técnico



Lucas Moreno, Jorge Díaz, Daniel Espinosa, Andrés Cabañas, Andrés Roldán

2º Desarrollo de Aplicaciones Multiplataformas 2022/23

Índice

1. Herramientas utilizadas durante el desarrollo	3
1.1. Github	3
1.2. Docker	3
1.3. Netbeans	3
1.4. JasperSoft	4
2. Desarrollo del código	4
2.1. InicioSesión	4
2.2. Registro	5
2.3. Agenda	6
2.3.1. AgendaPrincipal	6
2.3.2. AnyadirContacto	7
2.3.3. EditarContacto	7
2.4. Bloc de notas	7
2.4.1. BlogNotas	7
2.4.2. EditarNota	8
2.4.3. NuevaNota	8
2.5. ListaCompra	9
2.6. Cartera	10
2.6.1. AnyadirDinero	10
2.7. Informes	10

1. Herramientas utilizadas durante el desarrollo

1.1. Github

Al principio del proyecto elegimos esta herramienta para organizar las dependencias del trabajo de manera remota y cómodamente desde los distintos dispositivos de cada uno de los participantes del proyecto.

Al aplicar esta tecnología hemos optado por concederle la responsabilidad de cabeza de proyecto a uno de nuestros integrantes, el cual se ha encargado de manejar la plataforma de Github integrando los cambios de cada de rama y solucionar los errores de compatibilidad.

1.2. Docker


Docker ha sido la pieza central del proyecto mientras estábamos desarrollando la base de datos. Este nos ha permitido replicarla en cualquier dispositivo sin tener que tener un servidor remoto, ya que Docker permite alojar un contenedor de forma y con persistencia de datos.

Para esta parte del trabajo hemos comenzado creando un archivo “Docker-compose”, el cual duplica desde los servidores de Docker un contenedor SQLServer en el cual se procedió a crear la base de datos, junto con todas las tablas pertinentes y sus datos correspondientes.

1.3. Netbeans

Debido a qué el lenguaje que íbamos a utilizar durante todo el proyecto era Java, teníamos una baraja de opciones, de las cuales la más óptima según lo que teníamos que implementar en el trabajo, nos ofrecería mayor soltura.

Durante el desarrollo de nuestro proyecto, este programa ha sido la clave principal, porque era nuestro editor de código.



A la hora de trabajar optamos por una distribución en clases, la cual otorga una mayor compartimentación, para que sea más fácil la corrección de errores. Para el apartado visual optamos por usar JavaFX, puesto que es el principal motor visual en Java y otorga mucha facilidad para crear ventanas de interfaces responsives.

1.4. JasperSoft

JasperSoft pese a su antigüedad sigue siendo una de las principales herramientas a la hora de crear informes para aplicaciones Java, debido a que es de código abierto y tiene una comunidad muy amplia y con muchas respuestas a los usuales problemas que suele generar el programa debido a su implementación.

Para poder crear dichos informes, principalmente, hemos tenido que diseñarlos a mano y vincularlos a la base de datos. Acto seguido, hemos creado en nuestra aplicación Java, ciertas funcionalidades, las cuales con un solo click, permite generar los informes ya creados correspondientes al usuario que lo solicita.

2. Desarrollo del código

2.1. InicioSesión

Comenzando con la página inicial de nuestro proyecto tenemos el “InicioSesión”, este apartado consta de una única clase.

- En las primeras líneas del código, se encuentran los “imports” desde la línea 7 a la 11.
- Acto seguido, ya dentro de la clase principal, nos encontramos con la inicialización del apartado visual de la ventana actual.
- Después, están los distintos métodos que componen el programa, el primer de ellos es la acción en click del botón de registro, el cual te envía a la página de registro y cierra la anterior.
- Tras esto se halla la acción en click del botón de inicio de sesión, el cual se compone de una conexión a la base de datos, lo que permite comparar los datos introducidos en los campos de la aplicación con los registrados en nuestra base de datos, si los datos coinciden la aplicación procede a enviarte dentro de la aplicación, en caso contrario te mostrará un mensaje con el error.
- Seguidamente está el método usado para simular un “Placeholder”.
- Casi al final de la clase se ubica el método que crea y muestra el formato.
- Por último, tenemos las declaraciones de todos los objetos usados en esta pantalla.

2.2. Registro

Siguiendo con el proyecto tenemos el “Registro”, este apartado vuelve a constar de una única clase.

- En las primeras líneas del código, se encuentran los “imports”.
- Acto seguido, ya dentro de la clase principal, nos encontramos con la inicialización del apartado visual de la ventana actual.
- Posteriormente nos encontramos con el método en click de registrarse, en el cual se encuentra la validación de aquellos datos introducidos, así como el apartado de insertar el usuario en la base de datos y cambiar de pestaña al validar los datos insertados.
- Inmediatamente están los “MouseClicked” de los “TextFields”, los cuales tienen un “placeholder” que desaparece al clicar en ellos.
- Casi al final de la clase se ubica el método que crea y muestra el formato.
- Por último, tenemos las declaraciones de todos los objetos usados en esta pantalla.

2.3. Agenda

Este apartado consta de tres clases.

2.3.1. AgendaPrincipal

Esta pantalla, es la pantalla principal de las tres.

- Empezamos con los “imports” como en las demás pantallas.
- Acto seguido, se encuentra la iniciación de los componentes. Así como la conexión general de la pantalla actual y la carga de datos en la tabla principal.

- El siguiente método realiza la función de cambiar la pestaña a la pantalla de “AnydirContacto”.
- Seguidamente viene un menú principal de navegación que se comparte entre ventanas cuyos componentes son cinco botones, que permiten navegar en la aplicación.
- Después, tenemos el método “EditarContacto” que revisa la selección de la tabla, guarda los datos adquiridos y cambia la pestaña a la de “EditarContacto” con los valores seleccionados.
- Luego, tenemos el método “EliminarContacto” que revisa la selección de la tabla y los elimina.
- Casi al final de la clase se ubica el método que crea y muestra el formato.
- Por último, tenemos las declaraciones de todos los objetos usados en esta pantalla.

2.3.2. AnyadirContacto

- Comenzamos las primeras líneas con los “imports” correspondientes.
- Después, tenemos la iniciación de los componentes correspondientes, como la conexión a nuestra base de datos para poder añadir los contactos.
- El próximo método, lo que realiza es la acción de verificar que los datos del contacto añadido sean correctos y si son correctos, los inserta en la base de datos, si no manda un mensaje de error.
- Casi al final de la clase se ubica el método que crea y muestra el formato.
- Por último, tenemos las declaraciones de todos los objetos usados en esta pantalla.

2.3.3. EditarContacto

- Comenzamos las primeras líneas con los “imports” correspondientes.
- Después, tenemos la iniciación de los componentes correspondientes, como la conexión a nuestra base de datos para poder editar los contactos elegidos.
- El método más importante de esta clase, verifica que los datos editados son válidos y si lo son, los modifica en la base de datos y en la aplicación.
- Casi al final de la clase se ubica el método que crea y muestra el formato.
- Por último, tenemos las declaraciones de todos los objetos usados en esta pantalla.

2.4. Bloc de notas

Este apartado consta de tres clases.

2.4.1. BlogNotas

- Se comienza con los “imports” correspondientes de esta pantalla.
- Acto seguido, se encuentra la iniciación de los componentes. Así como la conexión general de la pantalla actual y la carga de datos en una tabla en la cual cada apartado es un botón.
- El siguiente método visible es el de añadir una nota nueva, este muestra la pantalla de “nueva nota”.
- Después, tenemos el método de abrir la nota, que sirve tanto para editar los datos como para visualizarlos.
- El último de estos métodos es el de eliminar la nota, borra los datos de la base de datos y de la tabla donde se encuentren los datos.
- Casi al final de la clase se ubica el método que crea y muestra el formato.
- Por último, tenemos las declaraciones de todos los objetos usados en esta pantalla.

2.4.2. EditarNota

- Comenzamos las primeras líneas con los “imports” correspondientes.
- Después, tenemos la iniciación de los componentes.
- El primer método que nos encontramos, es el boton de “reset”, que setea los campos a cadena vacía.
- El próximo método visible, es el “cancelar”, que cancela la edición de la nota.
- Luego, tenemos el método “EditarNota” en el que los datos insertados se vuelven los datos actuales en la nota abierta.
- Casi al final de la clase se ubica el método que crea y muestra el formato.
- Por último, tenemos las declaraciones de todos los objetos usados en esta pantalla.

2.4.3. NuevaNota

- Comenzamos las primeras líneas con los “imports” correspondientes.
- Después, tenemos la iniciación de los componentes.
- El primer método que nos encontramos, es el boton de “reset”, que setea los campos a cadena vacía.
- El próximo método visible, es el “cancelar”, que cancela la edición de la nota.
- Posteriormente tenemos el método de añadir una nota nueva, en el cual se insertan los datos en la base de datos.
- Casi al final de la clase se ubica el método que crea y muestra el formato.
- Por último, tenemos las declaraciones de todos los objetos usados en esta pantalla.

2.5. ListaCompra

- Empezamos con los “imports” como en las demás pantallas.
- Acto seguido, se encuentra la iniciación de los componentes. Así como la conexión general de la pantalla actual.
- El primer método que nos encontramos en esta pantalla, realiza la función de cargar aquellos artículos que el cliente ha elegido.
- El siguiente método, adquiere los productos que el cliente desea eliminar y los elimina tanto de la base de datos como del panel donde se encuentran visualmente los artículos. Aquí se eliminan solamente los productos que el cliente ha seleccionado para su eliminación.
- Después, está el método de añadir un producto, el cliente simplemente elige el artículo que quiere añadir y se añade a la base de datos y al panel de la aplicación.
- Como último método tenemos, aquel que borra todos los items, este método se utilizará cuando el cliente quiera borrar su lista de la compra en su totalidad.
- Casi al final de la clase se ubica el método que crea y muestra el formato.
- Por último, tenemos las declaraciones de todos los objetos usados en esta pantalla.

2.6. Cartera


Este apartado consta de una única clase.

2.6.1. AnyadirDinero

- Empezamos con los “imports” como en las demás pantallas.
- Acto seguido, se encuentra la iniciación de los componentes. Así como la conexión general de la pantalla actual.
- Primeramente, el método realiza la función de añadir la cantidad de dinero exacta que el cliente ha ingresado, todo esta masa de dinero en la cuenta y en la base de datos.
- Próximamente, tenemos el método contrario al primero, retira la cantidad de dinero exacta que el cliente quiere. Este se retira tanto de la base de datos como del panel visual en la aplicación.
- El último método, es el método que hace la suma o la resta del dinero, en base a lo que tenga el cliente en su cuenta bancaria.
- Casi al final de la clase se ubica el método que crea y muestra el formato.
- Por último, tenemos las declaraciones de todos los objetos usados en esta pantalla.

2.7. Informes

- Se comienza con los “imports” correspondientes de esta pantalla.
- Acto seguido, se encuentra la iniciación de los componentes. Así como la conexión general de la pantalla actual.
- El primer método, llama al primer reporte hecho en JasperSoft, el cual es la media de los contactos.
- El segundo método y último, llama al segundo reporte, que es las transacciones hechas por el cliente.

- 
- Casi al final de la clase se ubica el método que crea y muestra el formato.
 - Por último, tenemos las declaraciones de todos los objetos usados en esta pantalla.