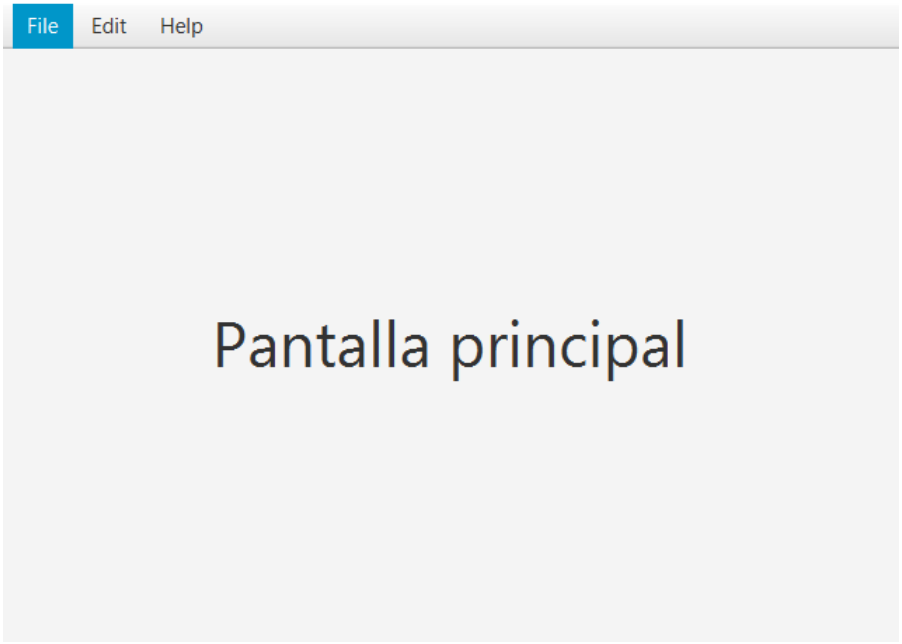
INSTRUCCIONES

- La práctica debe realizarse de manera individual y se dedicarán horas en clase para resolver posibles dudas que surjan.
- El contenido a entregar será un archivo comprimido con el proyecto de JavaFX incluyendo el nombre “Nombre_Apellido1_Apellido2_PrácticaUnidades1-2.zip”. Por ejemplo, “José_López_Pérez_PrácticaUnidades1-2.zip”.
- Se valorará que el nombre del proyecto de Eclipse tenga una nomenclatura similar a la anterior (por ejemplo, José_López_Pérez_PrácticaUnidades1-2).
- De la misma manera, el proyecto debe estar bien organizado empleando paquetes divididos por funcionalidad. Por ejemplo, para las vistas o controladores, o para las diferentes pantallas.
- **Es obligatorio subir las modificaciones gradualmente en la cuenta de GitHub creada el primer día de clase. Se puede crear una rama específica para la práctica.**
- La entrega de la práctica debe realizarse a través del aula virtual en la tarea asignada.

ENUNCIADO DE LA PRÁCTICA

El objetivo de la práctica es realizar el diseño (de momento sin funcionalidad) de una aplicación de gestión empresarial de un ámbito que tendrás que decidir.

La aplicación tendrá un menú principal con un formato similar al que se muestra a continuación:



La manera más sencilla de implementar un menú de estas características es empleando un **BorderPane** y llevando el menú a la parte superior y el contenido principal a la sección central, aunque puedes emplear una alternativa diferente o incluso cambiar la posición del menú.

El software de gestión a desarrollar tendrá inicialmente dos o tres pantallas como mínimo:

- Una pantalla donde se muestran datos correspondientes a una entidad asociada al ámbito de la empresa. Esta pantalla debe mostrar un listado o tabla con todos los datos disponibles hasta el momento. La entidad puede referirse a cualquier objeto susceptible de ser incluido en una base de datos. Por ejemplo: empleados, productos, cuentas bancarias, nóminas, etc. De momento puedes utilizar datos de prueba.
- Una pantalla de mantenimiento del listado anterior. Será como una especie de formulario que permite añadir, editar o eliminar registros. Puedes crear esta vista en una pantalla nueva, o también es posible en la pantalla mencionada anteriormente con el listado.
- Un tutorial de uso de la aplicación. Se trata de añadir una pantalla organizada de la forma más eficiente con diferentes apartados para elaborar un manual de ayuda al usuario. No hace falta que sea una pantalla muy extensa, sino que sirva para hacerse una idea de cómo sería la sección de ayuda.

Se cambiará entre las diferentes pantallas empleando el menú, de la forma que se explicará más adelante en un anexo al final del documento y con un proyecto de ejemplo que se incluye junto con el enunciado en el aula virtual.

La aplicación tendrá los siguientes requisitos sobre el uso de JavaFX:

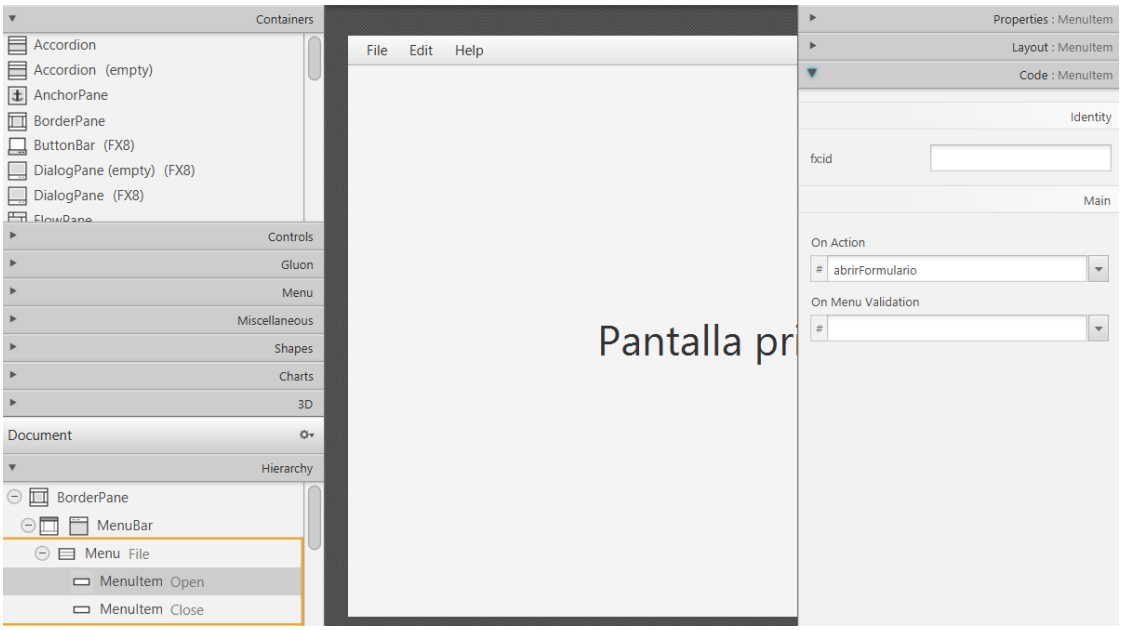
- Emplear al menos dos layouts para el contenedor principal de las pantallas usando los tipos **BorderPane**, **GridPane** o **AnchorPane.** Se pueden emplear alternativas más avanzadas como **SplitPane**, **TabPane**, **ScrollPane, Accordion** o **TitledPane**. En cualquier caso, el objetivo es poner en práctica como mínimo dos de los layouts mencionados.
- Emplear al menos tres layouts para posicionar elementos dentro de un contenedor de los tipos **HBox**, **VBox**, **FlowPane**, **TilePane** o **StackPane**.
- A la hora de diseñar la entidad con los datos de la empresa, se valorará imaginar datos lo suficientemente complejos que en impliquen añadir en la pantalla de mantenimiento una gran variedad de controles como **Button**, **Label**, **CheckBox**, **RadioButton**, **TextField**, **PasswordField**, **TextArea, Slider, ChoiceBox, ComboBox** o **ListView**.
- En la pantalla de visualización de los datos de la empresa o en el tutorial, debes emplear como mínimo tres controles de los tipos **ChoiceBox**, **ComboBox**, **ListView**, **TableView** o **TreeView**
- Puedes añadir otros elementos como imágenes o enlaces. Se valorará en cualquier caso el uso de controles lo más complejos posible.

Como hasta ahora solamente hemos estudiado cómo crear aplicaciones con una sola pantalla, a continuación se añade un epígrafe en el que se indica una forma sencilla de cambiar el contenido a mostrar en una aplicación gráfica de escritorio.

ANEXO: Cómo abrir una pantalla empleando el menú

La aplicación de la práctica tendrá varias opciones a las que se acceder a través del menú. En este apartado se explica cómo cambiar el contenido a mostrar en función de la opción seleccionada.

Para ello, tenemos que emplear el evento **"On Action"** que se estudiará más adelante. Se puede configurar desde **SceneBuilder** en la pestaña **"Code"** situada a la derecha. Accedemos a esta pestaña sobre el ítem de menú que queremos que se encargue de abrir una nueva pantalla. Ahí incluimos el nombre del método del controlador que se va a encargar de llevar a cabo las acciones asociadas al menú.



En el menú de **SceneBuilder** en la ruta **View -> Show Sample Controller Skeleton** se puede copiar el método que se implementará en el controlador.

Como se indicaba en el enunciado, la forma más sencilla de incluir un menú es en la parte superior de un **BorderPane**. El contenido principal se añadirá en la sección central. La alternativa más eficiente es dar la opción al controlador del menú de acceder a la ventana de la aplicación. Para ello, le pasamos el **BorderPane** con un método set que nos creamos manualmente en el controlador. Esto se hace desde la clase **Main** cuando se crea la ventana principal con los objetos **Scene** y **Stage**.

```
// Pasamos al controlador de menú el objeto con el BorderPane principal
MenuController menuController = loader.getController();
menuController.setRootLayout(rootLayout);
```

Una vez en el controlador de la pantalla principal, basta con cargar el FXML con la pantalla que queremos mostrar y se mostrará en la sección central del **BorderPane** con **setCenter**. Y así sucesivamente con todas las pantallas. La idea es sobrescribir el contenido central por la pantalla que corresponda en cada momento.

```
@FXML
private void abrirFormulario(ActionEvent event) {
   try {
       // Cargamos el archivo Controles Dinámicos
       FXMLLoader loader = new FXMLLoader();

       loader.setLocation(MenuController.class.getResource("/basicoDinamico
             /ControlesDinamicos.fxml"));

       GridPane listadoControles = (GridPane) loader.load();

       // Se sitúa en el centro del diseño principal
       rootLayout.setCenter(listadoControles);
   } catch (IOException e) {
       e.printStackTrace();
  }
}
```

Podemos crear métodos similares a **abrirFormulario** con tantas pantallas como queramos.

Por otro lado, una alternativa para dejar el contenido centrar en blanco sería pasarle el valor **null.**

```
@FXML
private void cerrarListado(ActionEvent event) {
   // Se elimina el contenido del nodo central
   rootLayout.setCenter(null);
}
```

Cabe reiterar que se adjunta al enunciado un proyecto de JavaFX de ejemplo con una aplicación que cambia de pantalla al hacer click en los menús “File -> Open” y “File -> Close”. Puedes observar más detenidamente el código para ver el comportamiento.