Ejemplo guiado: formulario de login

El objetivo del ejercicio es diseñar con Scene Builder una interfaz de usuario que simule el login de un usuario en una aplicación. Utilizaremos un manejador de eventos para mostrar en pantalla un texto de bienvenida cada vez que se pulsa el botón Iniciar del formulario.

La apariencia final del formulario es la que se muestra en la Figura 1.

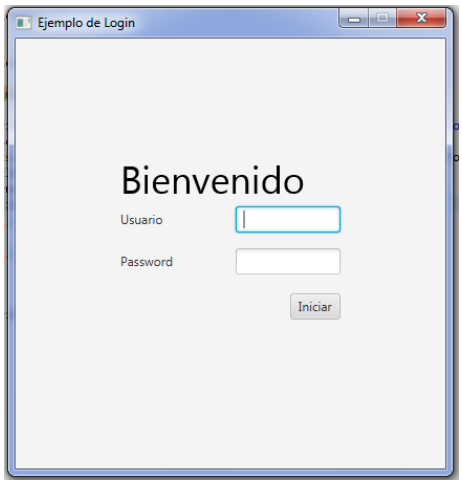

Figura 1.Interfaz Final

1. Crear el nuevo proyecto JavaFX 8 con Netbeans

(Si estas utilizando Java 11 en este punto debes de revisar el anexo 1)

Dentro del entorno NetBeans utilice *File -> NewProject* y seleccione un proyecto *JavaFX FXML Application*

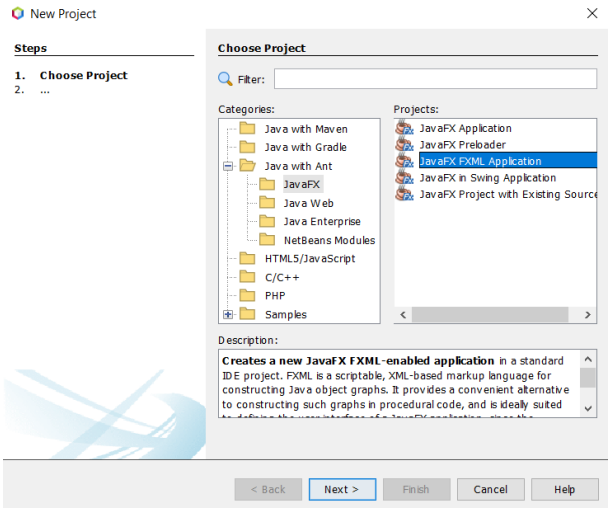

Figura 2. Tipos de proyecto

Pulse el botón *Next* y en la siguiente pantalla ponga un nombre al proyecto, *EjercicioLogin*. Cambie el nombre por defecto del fichero FXML *FXMLDocument* por *FXMLLogin* y después pulse *Finish.*

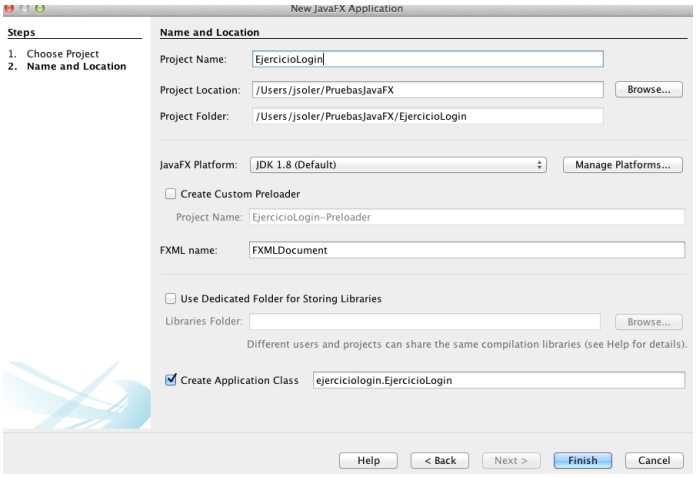

Figura 3. Propiedades del proyecto a crear

NetBeans ha creado la estructura del proyecto, tal como se muestra en la Figura 4.

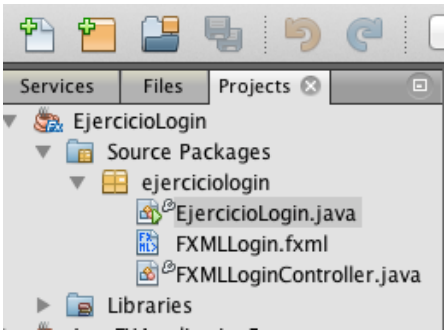

Figura 4. Estructura el proyecto.

Por defecto Netbeans ha creado el fichero XML con un *Anchorpane* que contiene *un Button* y un *Label* . En la Figuras 5, Figura 6 y Figura 7 puede verse el contenido de los ficheros del proyecto creado por el asistente.

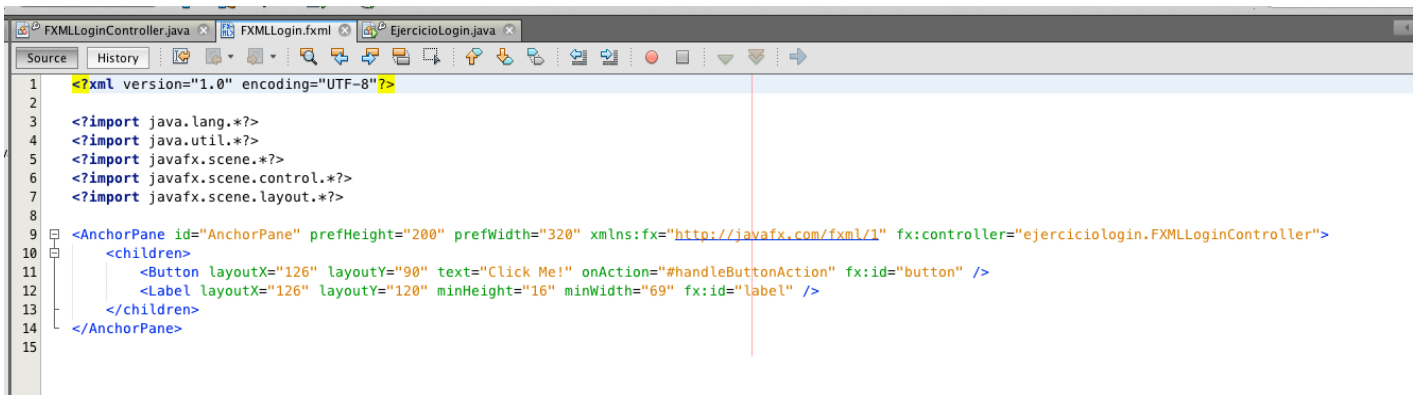

Figura 5. Contenido del fichero FXML de la pantalla principal de un proyecto creado con el asistente.

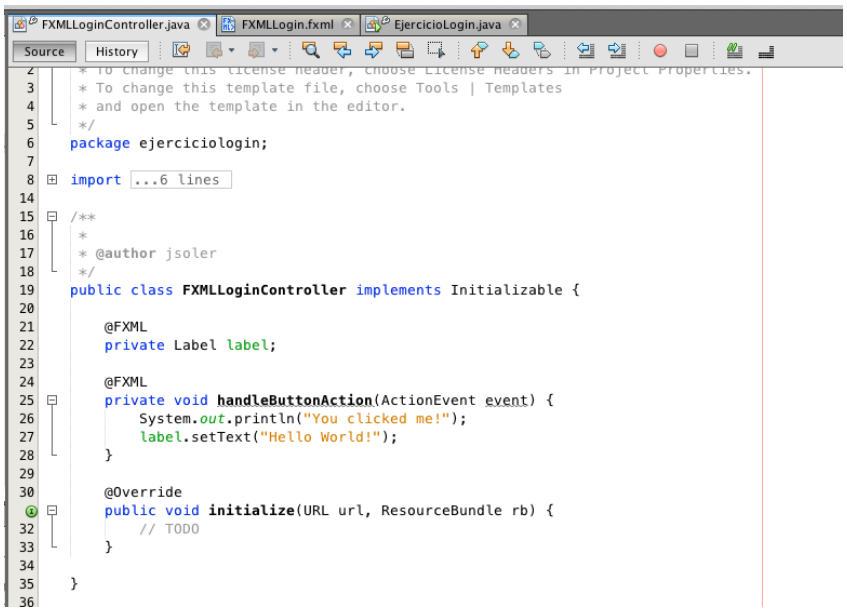

Figura 6. Código de la clase de Java "Controlador"

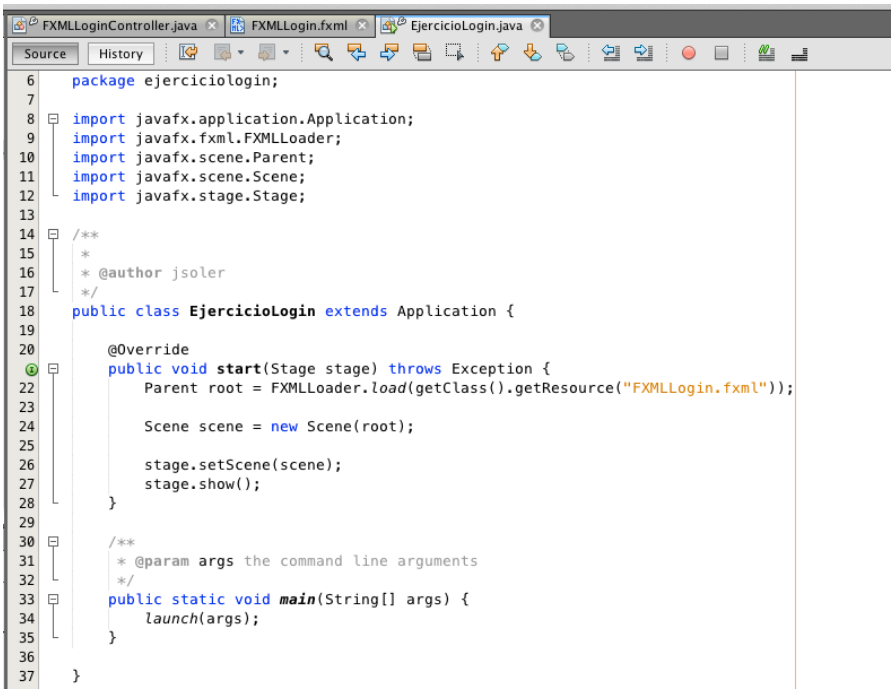

Figura 7. Código de la clase principal de la aplicación

Podemos ejecutar el proyecto creado con la opción de menú *Run Project.*

2. Crear la interfaz con Scene Builder

Para crear la interfaz vamos a reutilizar el fichero FXML generado por defecto. El fichero FXMLLoginController.java no nos va a resultar útil, antes de seguir lo eliminaremos del explorador del proyecto.

Ahora diseñaremos la interfaz de usuario, para ello marcamos el archivo FXMLLogin.fxml y con el botón derecho del ratón en el menú contextual *Open* (ó simplemente doble click)

La Figura 8 muestra la interfaz de usuario de Scene Builder:

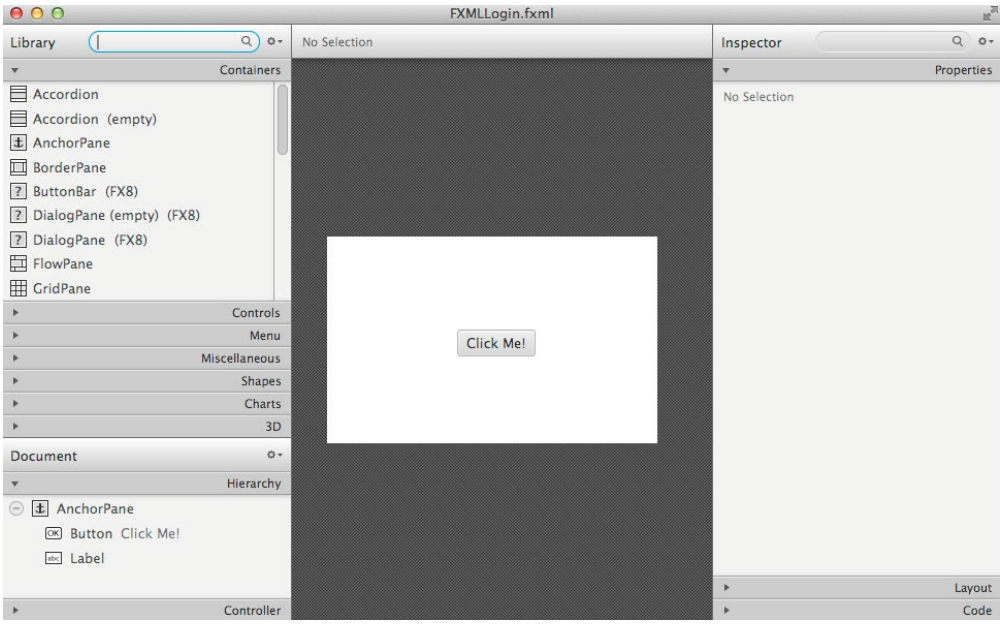

Figura 8. Scene Builder

Ahora vamos a eliminar todos los elementos gráficos del fichero (AnchorPane, Button, Label) y empezaremos a diseñar la interfaz de login. Para ello, en el panel de la izquierda, dentro de la sección Document, selecciona el AnchorPane actual y presiona Suprimir.

Vamos a empezar añadiendo un layout *GridPane* que representa una matriz de filas y columnas en las cuales se pueden situar componentes de la interfaz de usuario. Definiremos una matriz de 3 filas y 2 columnas (2x3). Para ello selecciona un GridPane de la librería de controles y arrástralo a la zona de trabajo. Por defecto se creará un grid de 2x3.

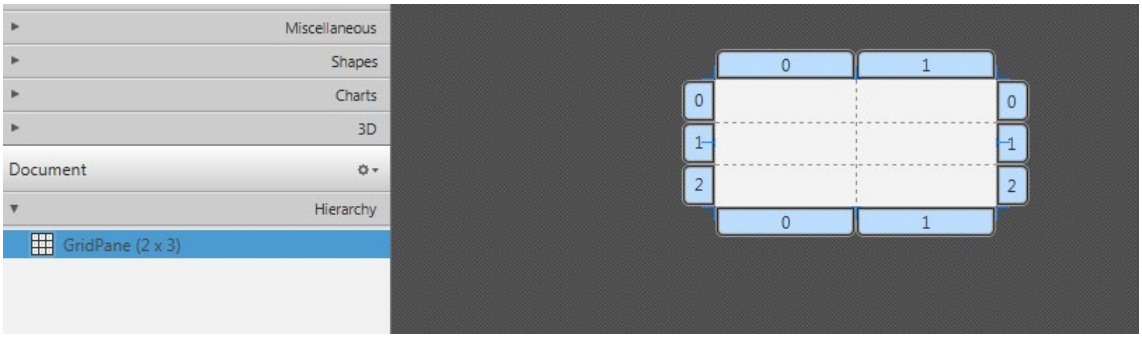

Figura 9. GridPane

Si necesitamos más filas o columnas, sobre el panel izquierdo *Hierarchy* y usando el menú contextual (el botón derecho del ratón) *nos aparecerán todas las opciones necesarias*. De la misma manera, en la zona de trabajo podemos seleccionar una fila o columna y con el botón derecho del ratón nos aparecerán las opciones para modificar el GridPane (ver Figura 10).

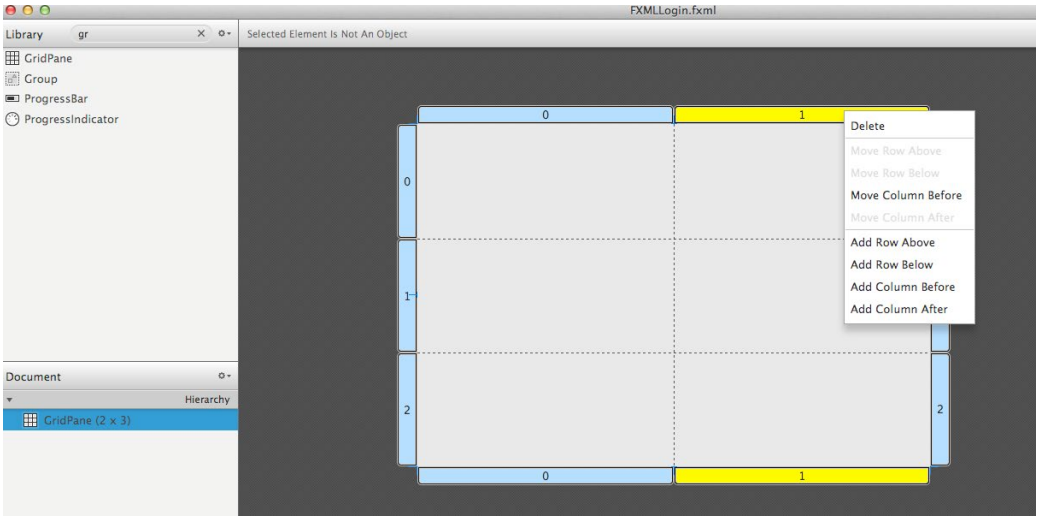

Figura 11. Modificar las características del GridPane

El objetivo ahora es diseñar la ventana mostrada en la Figura 11.

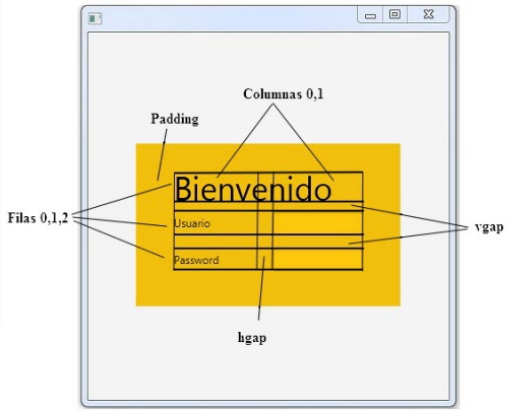

Figura 10. Ventana a diseñar

Empezaremos añadiendo desde Scene Builder algunos de los componentes de la interfaz de usuario. Usaremos un componente *Text* para el mensaje de bienvenida y dos componentes *Label* para las etiquetas *usuario* y *password*. Los seleccionamos en la paleta (puede usar la función de búsqueda en Library, tal y como se indica en la Figura 12) y los dejamos respectivamente en las tres filas de la matriz. Escriba en la propiedad Text de cada uno de los componentes: Bienvenido, Usuario y Password.

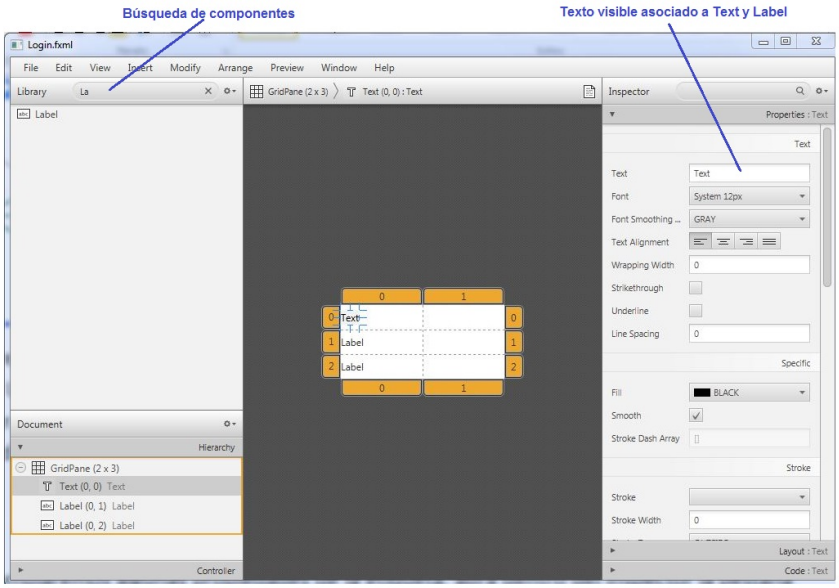

Figura 12. Añadir los primeros componentes

Ahora cambiaremos algunas propiedades del componente Tex (Bienvenido). Abra dentro del inspector la pestaña *Layout* y escriba en *Columspan* 2 (Figura 13). Esto permite que el componente pueda ocupar, si es necesario, la segunda columna.

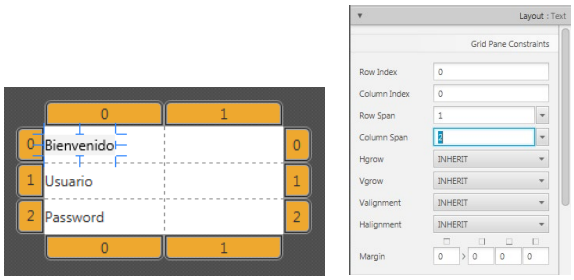

Figura 13. Cambiar las propiedades de un componente text

Ahora definiremos la separación entre filas (Vgap) y entre columnas (Hgap). Seleccione el GridPane y en Layout teclee los valores 10, 10 en Vgap y Hgap. Fijaremos también la el margen interno (padding) del gridPane. En Padding introduzca 25,25,10,25. El aspecto del componente es el de la Figura 14.

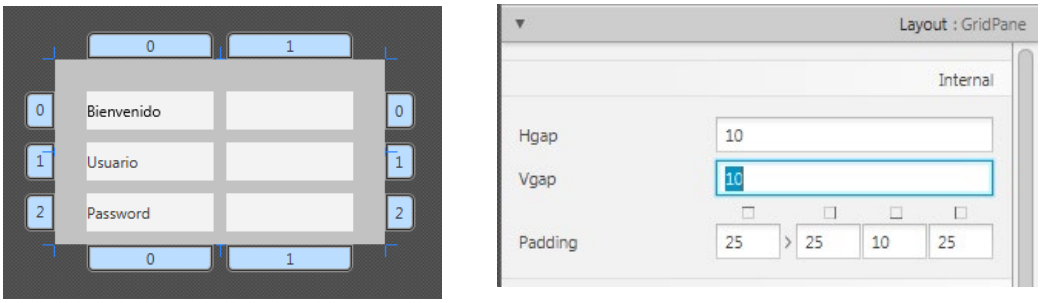

Figura 14. Aspecto del componente

Añadiremos ahora un componente para la entrada de texto (*TextField*) y otro para la entrada del password (*PasswordField*). Los arrastramos a sus respectivas posiciones. En Preview puede ver el aspecto de la interfaz (Figura 15).

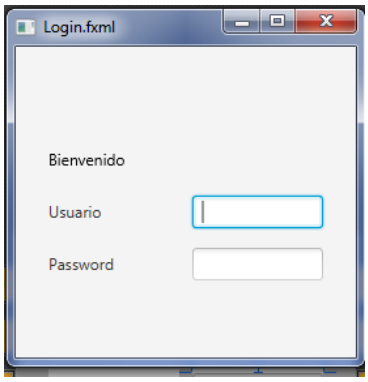

Figura 15. Preview del aspecto de la interfaz

Vamos a cambiar el tamaño de la fuente para el mensaje Bienvenido, de modo que pueda ocupar las dos columnas. Seleccione el componente Text y cambie en Properties -> Font el valor a 36.

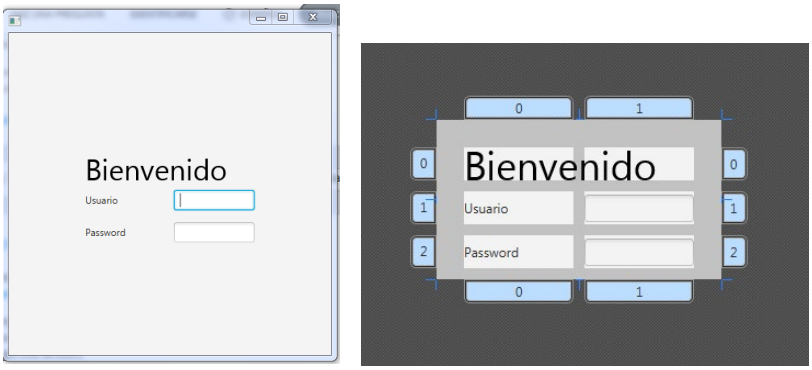

Figura 16. Cambiar el tamaño de la fuente del mensaje "Bienvenido"

Lo último que queda por añadir es un botón y el texto que muestra el inicio de sesión. Añada una nueva fila al final de la matriz y sitúe dentro en la posición 1,3 un panel HBox y dentro de éste un botón. Ponga el alineamiento del panel a Bottom-Right y cambie el texto del botón a Iniciar. Para el mensaje que recibirá el usuario utilizaremos en componente Text que se ubicará en una nueva fila de la matriz. Sitúe el componente Text el la primera columna y última fila,

cambie la propiedad Fill a color rojo, use pare eso la paleta de colores(). Además, cambie la propiedad Text del componente Text al string vacío.

Figura 17. Añadir el botón y el mensaje de inicio de sesión

Acciones necesarias para añadir el comportamiento

Para que el texto cambie cuando se ejecuta el programa y después de pulsar el botón *Iniciar*, el componente Text debe tener un ID, que luego será referenciado desde la clase controladora de la interfaz de usuario. Para ello seleccione el componente y en la pestaña Code escriba en el campo fx:id, debajo de Identity, ***mensaje_usuario***.

Figura 18. Identificador del componente Text

Vamos a hacer lo mismo en el TextField en el que se introduce el usuario, pondremos en el campo fx:id ***texto_usuario***

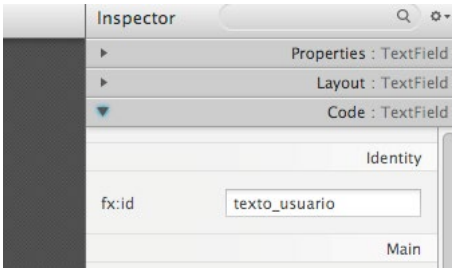

Figura 19. Identificador del TextField

Para finalizar añadiremos funcionalidad al botón, definiendo un manejador de evento que se ejecutará cuando éste resulte pulsado.

Seleccione el botón Iniciar y abra la pestaña del Inspector Code y busque *On Action*, incluya allí un nombre descriptivo como por ejemplo ***pulsadoIniciar***. Esto será el nombre de método que posteriormente tiene que ser implementado en la clase controlador.

Figura 20. Asociar un manejador en Scene Builder y en el fichero FXML

La figura anterior muestra el efecto en el archivo fxml de incluir el nombre del manejador de evento *pulsadoIniciar*.

Antes de abandonar SceneBuilder es necesario salvar el fichero con el que estamos trabajando.

3. Crear la clase de Java que se asocia como controlador

El código de manejo del evento lo situamos en la clase controladora que vamos a crear de manera automatica en NetBeans.

Para ello seleccionaremos en el explorador del proyecto el fichero FXMLLogin y con el boton derecho de ratón invocaremos a Make Controller

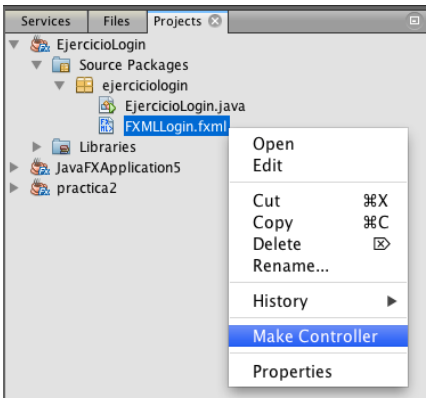

Figura 21. Generación automática de la clase controlador

Se crea automaticasmente el fichero FXMLLoginController, si ya existe se actualiza con los cambios introducidos en el fichero FXML. En la Figura 22 se muestra el resultado.

```
6   package ejerciciologin;
7
8   import ...7 lines
15
16  /**
17   * FXML Controller class
18   *
19   * @author jsoler
20   */
21  public class FXMLLoginController implements Initializable {
22
23      @FXML
24      private TextField texto_usuario;
25      @FXML
26      private Text mensaje_usuario;
27
28      /**
29       * Initializes the controller class.
30       */
31      @Override
32      public void initialize(URL url, ResourceBundle rb) {
33          // TODO
34      }
35
36      @FXML
37      private void pulsadoIniciar(ActionEvent event) {
38      }
39
```

Figura 22. Código generado

Como veis aparecen debajo de las anotación @FXML los elementos que en el fichero FXML tienen asignado un valor en la campo fx:id asi como el método que hemos indicado en el campo onAction del boton.

Para que la aplicación reaccione al click del boton tendremos que añadir la siguiente linea de código que aparece en la siguiente figura dentro del metodo pulsadoIniciar:

```
35
36      @FXML
37      private void pulsadoIniciar(ActionEvent event) {
38          mensaje_usuario.setText("Bienvenido "+ texto_usuario.getText());
39      }
40
```

Figura 23. Código del manejador de eventos

Ahora podemos ejecutar la aplicación desde NetBeans mediante *Run Project*

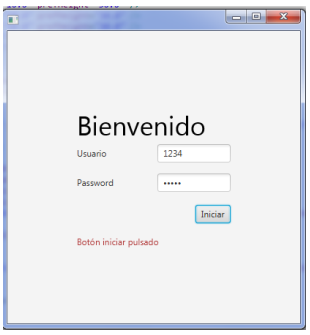

Figura 24. Resultado Final

Si queremos que la ventana tenga un título tenemos que añadir el siguiente código `stage.setTitle("Login")` en la clase Main.java (Figura 25). La instrucción `stage.setResizable(false)` impide el redimensionamiento del formulario.

```
20      @Override
21      public void start(Stage stage) throws Exception {
22          Parent root = FXMLLoader.load(getClass().getResource("FXMLLogin.fxml"));
23
24          Scene scene = new Scene(root);
25
26          stage.setScene(scene);
27
28          stage.setTitle("Login");
29          stage.setResizable(false);
30
31          stage.show();
32      }
```

Figura 25. Cambios en la clase principal

ANEXO I: Crear el nuevo proyecto JavaFX 11 con Netbeans

Netbeans nos permite crear un proyecto no modular mediante la herramienta Ant y un proyecto modular mediante Maven o Graadle. En nuestro caso vamos a optar por utilizar Ant. Podéis encontrar toda la información en el enlace https://openjfx.io/openjfx-docs/

Ant no dispone de un perfil específico para generar aplicaciones JavaFX 11 por lo que vamos a tener que crear un proyecto normal de Java y añadir las librerias necesarias. Para que podamos compartir los proyectos y poder realizar tutorías será necesario que la librería de JavaFX 11 que tenéis que definir tenga el mismo nombre, es por ello que proponemos definir la librería con el nombre: JAVAFX11

Antes de definir la librería en netbean tenemos que descargar en nuestra maquina el SDK de JavaFX: https://gluonhq.com/products/javafx/. Descomprime el fichero zip descargado.

En NetBeans añadiremos nuestra librería en el menú *Tools>Libraries* pulsaremos *New Library,* tal y como aparece en la siguiente figura.

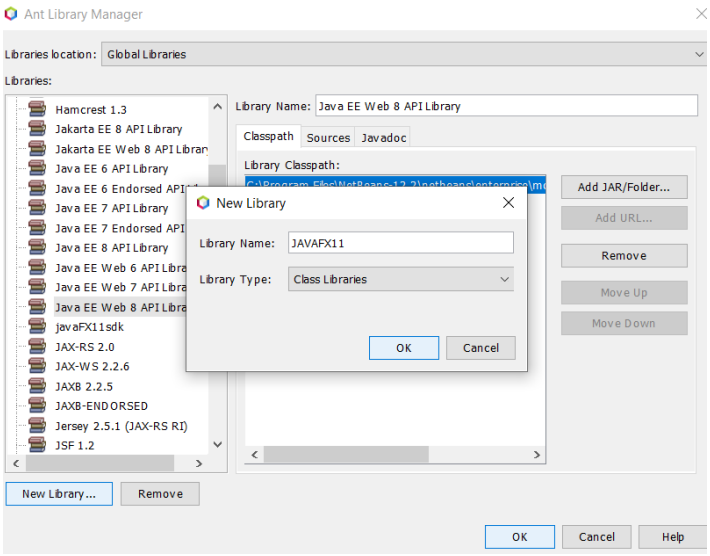

Figura 26. Añadir la librería JAVAFX11

Después de crear la librería, añadiremos a la misma los ficheros jar que la componen. Para ello, pulsaremos *Add JAR/Folder* y añadiremos todos los ficheros .jar que se encuentran en la carpeta lib del SDK de JavaFX que hemos descargado. **No incluyas** el fichero src.zip (ver Figura 27).

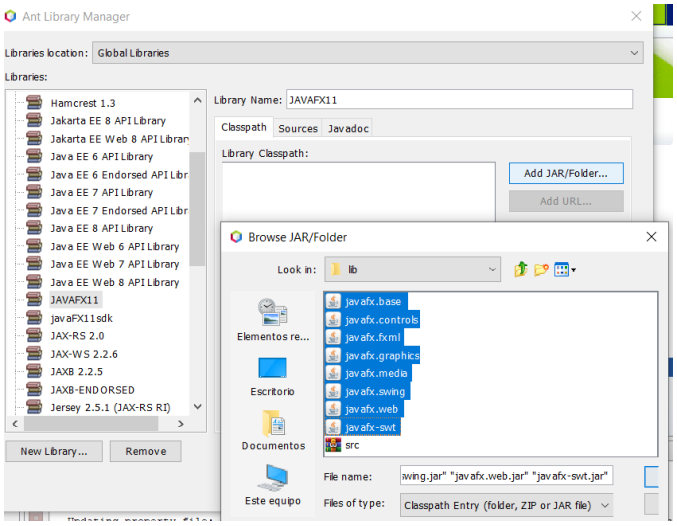

Figura 28. Añadir los jar de JavaFX a la librería

Ya tenemos la librería disponible para poder añadirla a un proyecto de Java. El proceso de inicialización siempre va a ser el mismo, para agilizarlo os proponemos que utilicéis un proyecto base ya configurado al que le podéis cambia el nombre. Puedes encontrar este proyecto base en Poliformat, dentro de la carpeta de prácticas, en un fichero zip comprimido llamado: JavaFXMLApplication.zip. Tras descomprimir, renombra la carpeta y déjala en la dirección que desees. En la siguiente figura puedes ver el contenido del proyecto:

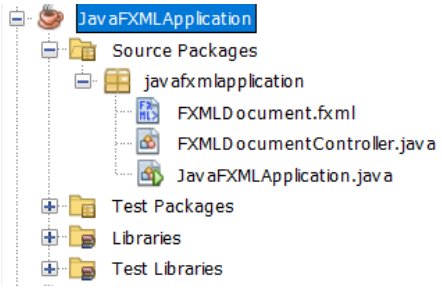

Figura 27. Estructura y ficheros del proyecto

Como puedes observar en la figura, el proyecto base se llama **JavaFXMLApplication**. Las modificaciones que hemos añadido a la configuración del proyecto son:

- En la ventana de configuración del proyecto, en el apartado de *libraries,* hemos añadido en el panel de compilación nuestra librería en el *Classpath*, y en el panel run en el *Modulepath*
- Por último hemos añadido en el apartado *run*, en *VM options:* **--add-modules javafx.controls,javafx.fxml**

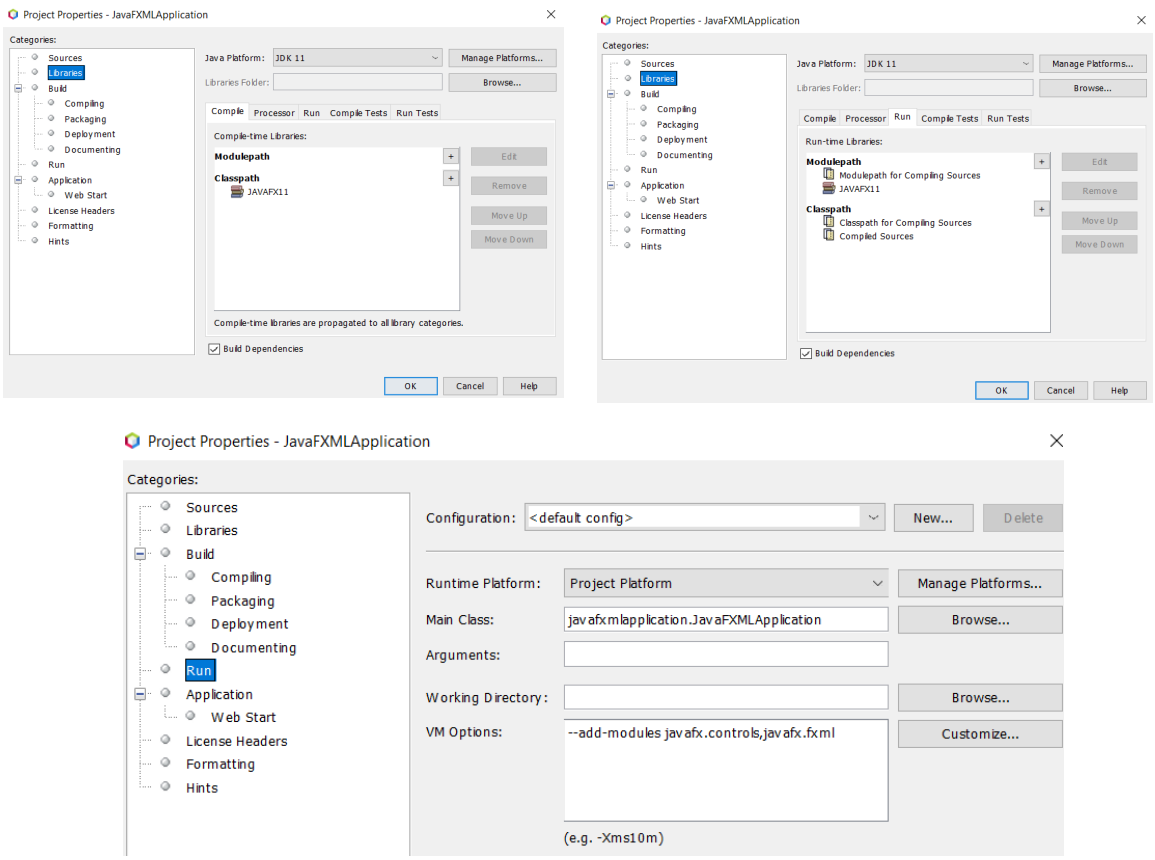

Figura 29. Modificaciones realizadas al proyecto en los apartados *Libraries* y *Run*

Por otra parte, todo proyecto Java se configura sobre una JDK definido en Netbeans, si el nombre de este JDK no coincide con el que tienes en tu maquina aparecerá un error al cargar el proyecto, simplemente ve a las propiedades del proyecto para resolver el problema seleccionando la plataforma adecuada dentro del apartado *Run*.

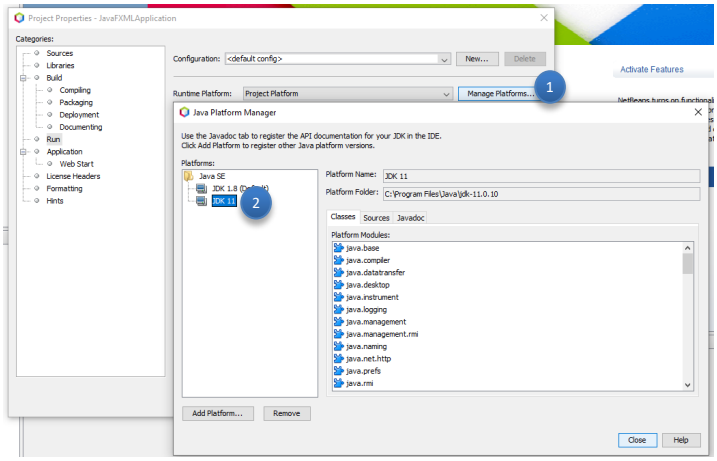

Figura 30. Seleccionar la plataforma de ejecución Java11

Una vez abras el proyecto **JavaFXMLApplication** en Netbeans, procede a cambiar el nombre del proyecto y los ficheros de forma que hagan referencia al proyecto que vayas a crear. Utiliza siempre que sea posible la opción *Refactor (*refactorizar).

Al utilizar *Refactor* para cambiar el nombre de los ficheros, del proyecto o de los packages, debes de tener en cuenta que esta opción **no actualiza** los cambios en los ficheros que no son .java. En particular debes de ser consciente que dentro de los ficheros .fxml hay una referencia a la clase de java (clase controladora) que acompaña a este fichero y que siempre deberás de modificar tu manualmente. Puedes editar directamente el fichero fxml cambiando la línea donde aparece el nombre de la antigua clase controladora por el nuevo nombre (en la siguiente figura, habría que cambiar el contenido de la línea 11).

```
8
9   <AnchorPane id="AnchorPane" prefHeight="200" prefWidth="320"
10              xmlns:fx="http://javafx.com/fxml/1"
                fx:controller="javafxmlapplication.FXMLDocumentController">
12      <children>
```

Figura 31. Línea actualizar con el nuevo nombre de la clase controladora de la escena diseñada en el fichero fxml

También puedes actualizar el nombre de la clase controladora desde SenceBuilder, seleccionando el nuevo nombre en la pestaña *Controller* situada debajo del panel izquierdo, tal y como se muestra en la figura Figura 32.

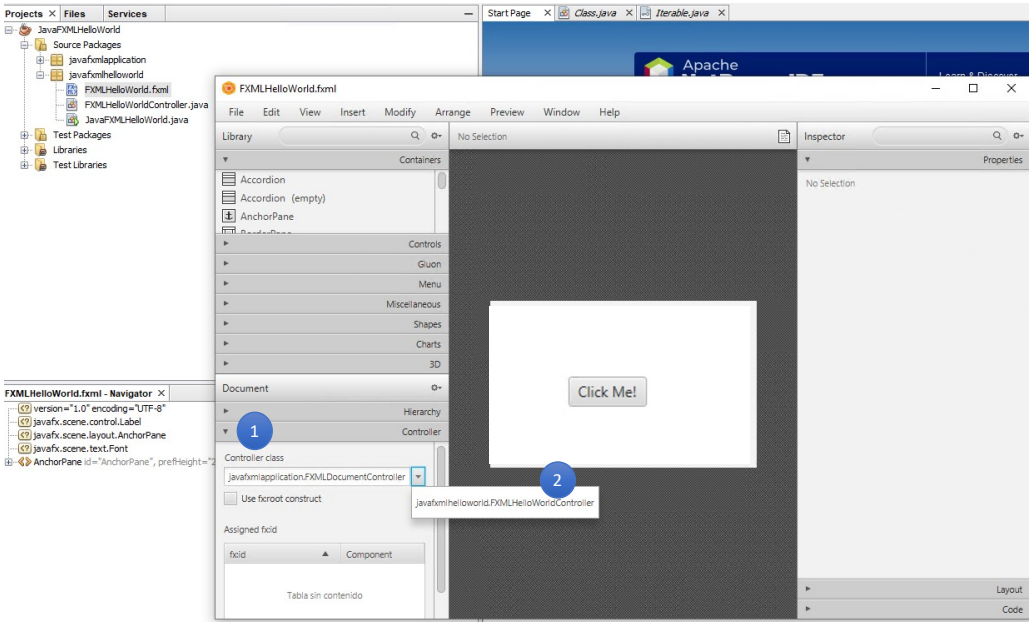

Figura 32. usar Scene Builder para actualizar la clase controladora de la ventana