

## Ejemplo guiado: Formulario de login

El objetivo del ejercicio es crear un proyecto de javaFX con Netbeans y diseñar con Scene Builder una interfaz de usuario que simule el formulario de login de un usuario en una aplicación. La apariencia final del formulario es la que se muestra en la Figura 1. Para hacer la aplicación más veraz le vamos a añadir funcionalidad, cuando el usuario pulse sobre el botón “Iniciar” la aplicación mostrará por pantalla un texto de bienvenida.

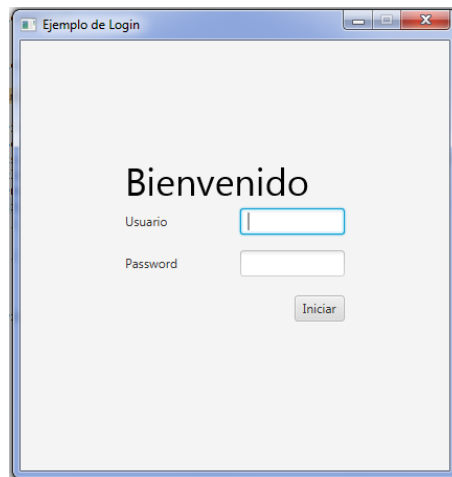


Figura 1. Interfaz Final

Para la realización de la práctica vamos a seguir los siguientes pasos: primero vamos a crear una aplicación JavaFX desde NetBeans utilizando la herramienta Ant, después diseñaremos la interfaz con SceneBuilder y por último añadiremos el código de java que da soporte a la funcionalidad. Al final del documento tienes los anexos con la información necesaria para poder configurar tu ordenador personal y poder realizar este ejercicio. Si no tienes preparado el portátil utiliza polilabs, la instalación del DSIC en windows.

### 1. Crear el proyecto JavaFX 11 con Netbeans

Como ya hemos indicado en la introducción a JavaFX, una aplicación javafx necesita arrancar en la función **main()** dentro de una clase que extienda la clase **Application**, además es necesario añadir la librerías propias de JavaFX. Este proceso se repite en cualquier aplicación de este tipo y es por ello por lo que es interesante automatizar el proceso. Desde Netbeans podemos crear un proyecto de JavaFX de varias maneras, mediante la herramienta Ant (en este caso no tenemos manera de automatizar el proceso), o mediante Maven y Gradle, (permiten la automatización).

Dada la instalación de los laboratorios del DSIC, resulta interesante utilizar Ant. Para evitar el inconveniente de la falta de automatización en la creación del proyecto vamos a seguir una estrategia ad hoc, en lugar de crear un proyecto nuevo para cada ejercicio vamos a abrir un

proyecto base existente, y después renombraremos el proyecto. Se aconseja utilizar un nombre acorde al ejercicio.

Descarga el proyecto base desde el siguiente enlace: [https://github.com/jsolerb/IPC\\_FXMLCore](https://github.com/jsolerb/IPC_FXMLCore).

El proyecto también está disponible en poliformat:

[https://poliformat.upv.es/access/content/group/GRA\\_11556\\_2021/Castellano/Pr%C3%A1cticas/IPC\\_FXMLCore-master.zip](https://poliformat.upv.es/access/content/group/GRA_11556_2021/Castellano/Pr%C3%A1cticas/IPC_FXMLCore-master.zip)

Después de descomprimir el proyecto lo tienes que abrir en NetBeans desde la opción *Open Project*:

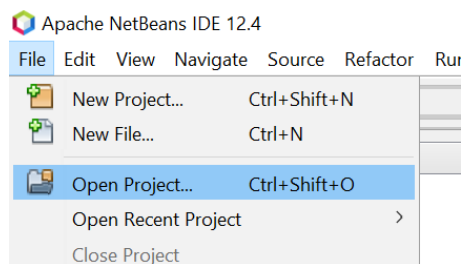


Figura 2. Open Project

Este proyecto está creado con la configuración del laboratorio (polilabs), los que utilizáis para realizar las prácticas vuestro propio ordenador tenéis que configurar Netbeans y crear una librería con los ficheros de javaFX. El proceso para seguir está en [JAVAFX11](#). Cuando definas la librería debería de desaparecer el error en el proyecto

Si tienes instalados en tu maquina varios JDK o el nombre de la instalación no coincide con la del laboratorio te aparecerá un error. Netbeans te dirige para resolver el error, en este caso simplemente selecciona del desplegable el nombre de tu plataforma JDK y el problema estará resuelto.

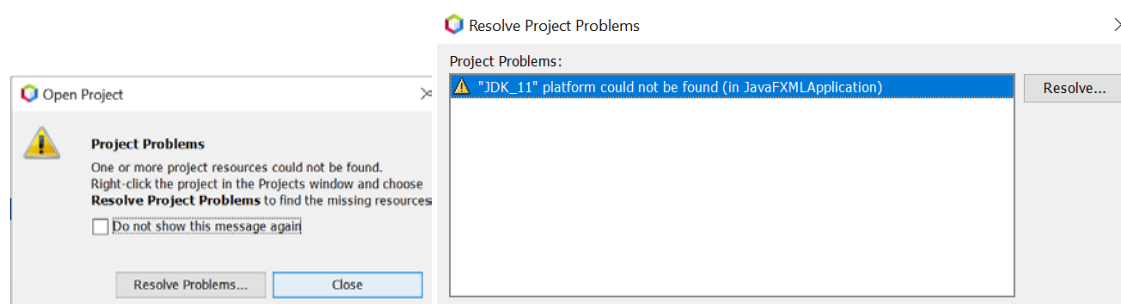


Figura 3. Problema con el nombre de la plataforma JDK

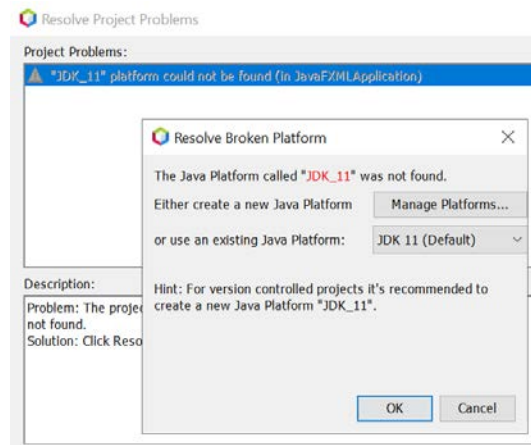


Figura 4. Asignar plataforma JDK al proyecto

Como puedes observar el proyecto base se llama **IPC\_FXMLCore** y dentro del paquete `javafxmlapplication` tienes los tres ficheros que componen el proyecto.

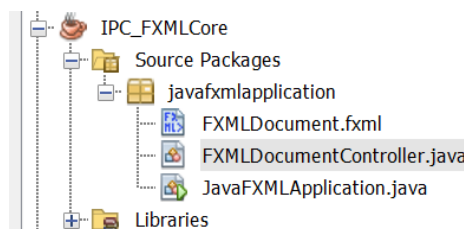


Figura 5. Estructura del proyecto

Ahora vamos a renombrar el proyecto para realizar el ejercicio. Selecciona el proyecto y pulsando el botón derecho sobre el proyecto en el explorador de Netbeans, aparece la opción de menú *Rename*:

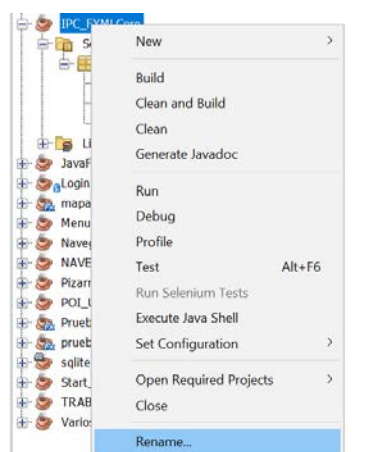


Figura 6. Opción Rename

Introduce el nuevo nombre y marca la casilla que te permite cambiar también el nombre de la carpeta donde está el proyecto.

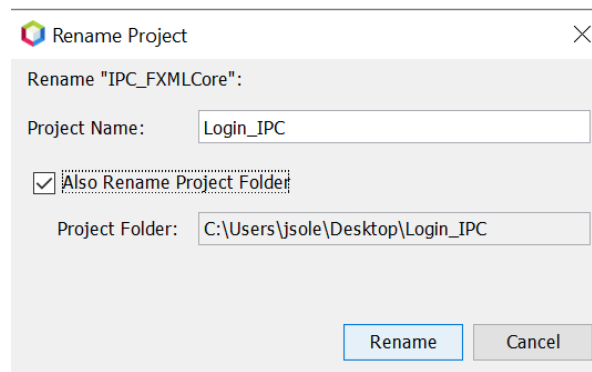



Figura 7. Formulario para Rename

Si todo ha ido bien deberías de poder ejecutar el proyecto, para ello tienes que pulsar sobre el icono play  que está en la barra de herramientas o con la opción *Run Project* del menú *Run*.

## 2. Crear la interfaz con Scene Builder

Para crear la interfaz vamos a editar el fichero FXMLDocument.fxml del proyecto, si pulsamos sobre el proyecto con el botón derecho nos aparece el menú contextual y tenemos las opciones *Open* que abrirá el proyecto con el editor SceneBuilder (también doble click) y la opción *Edit* que abre el fichero en modo texto dentro de Netbeans.

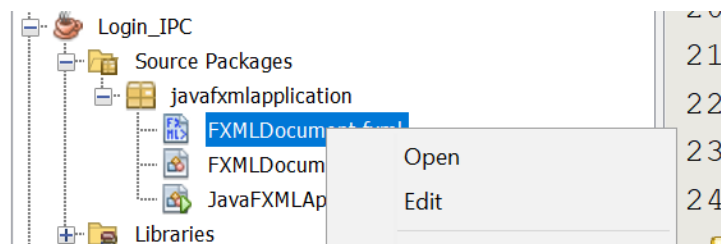


Figura 8 Open el fichero FXML en SceneBuilder

La Figura 9 muestra la interfaz de usuario de Scene Builder. Ahora vamos a eliminar todos los elementos gráficos del fichero y empezaremos a diseñar la interfaz de login desde cero. Para ello, en el panel de la izquierda, dentro de la sección Document > Hierarchy, selecciona el VBox que está en la raíz del grafo de escena y presiona *Suprimir*.

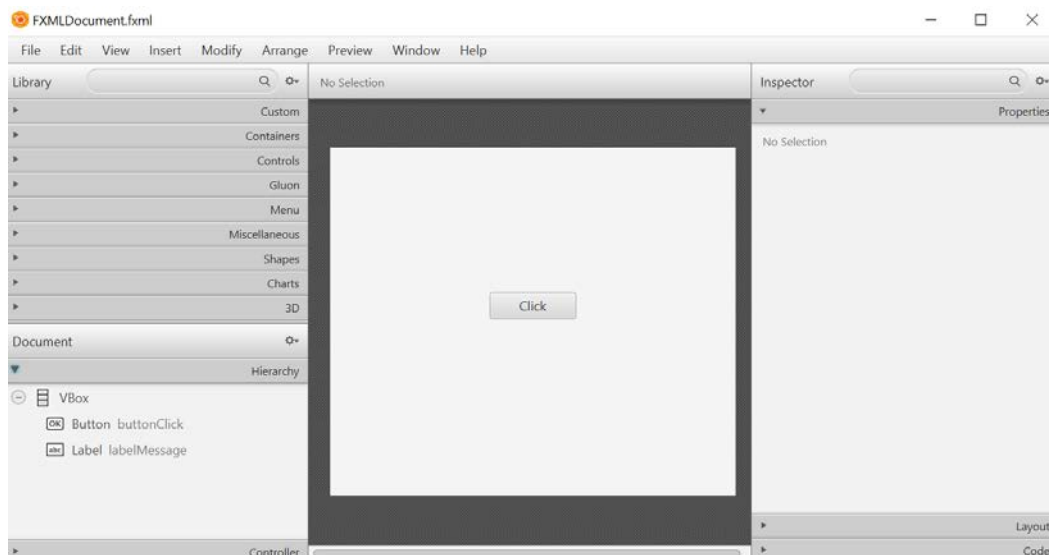


Figura 9 SceneBuilder

Vamos a empezar añadiendo un contenedor *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.

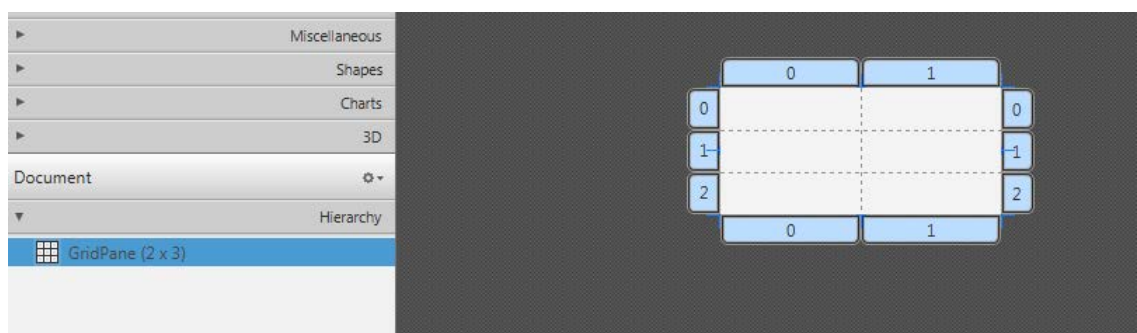


Figura 10 Gridpane

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*, Figura 11

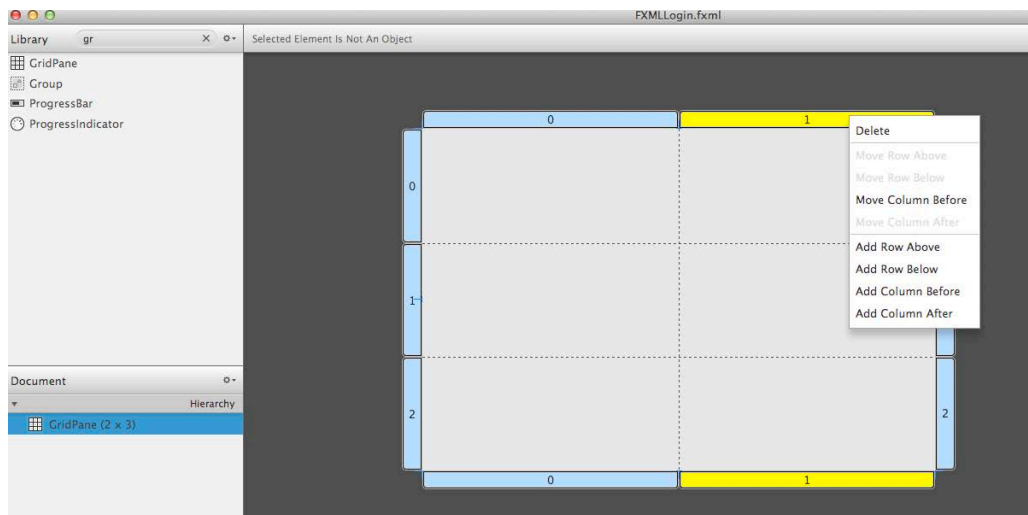


Figura 11 Añadir fila y columnas al gridpane

El objetivo ahora es diseñar la ventana mostrada en la Figura 1. 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) 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.

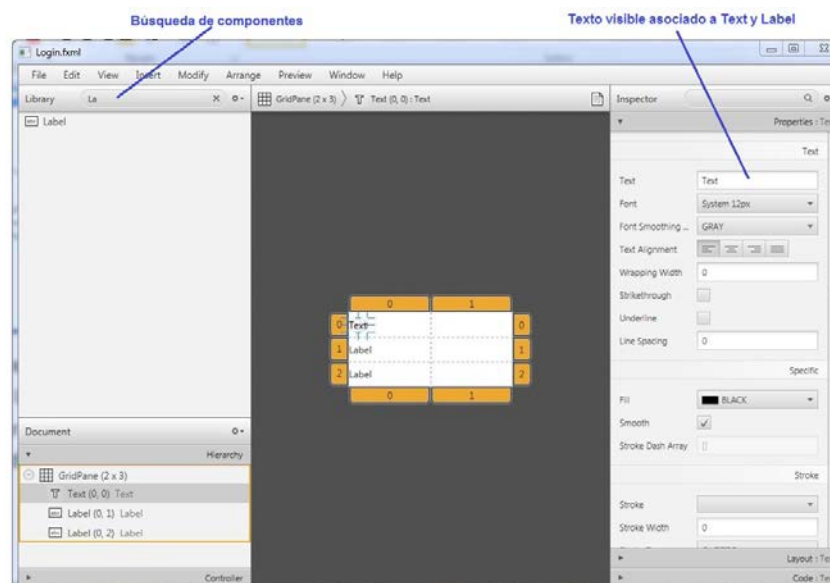


Figura 12

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

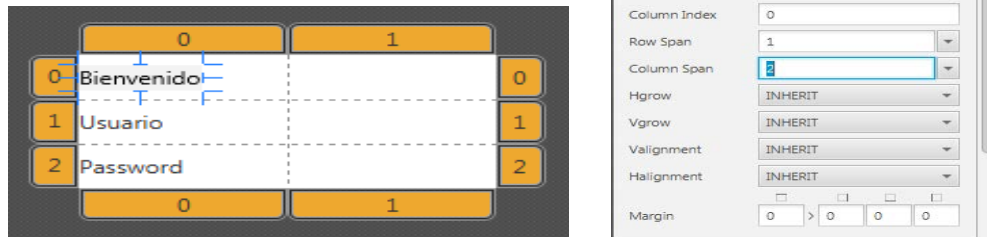


Figura 13

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 el margen interno (padding) del gridPane. En Padding introduzca 25,25,10,25. El aspecto del componente es el de la figura 14.

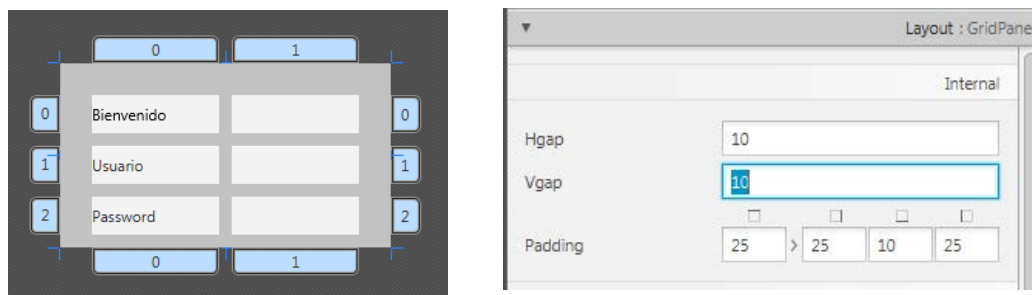


Figura 14

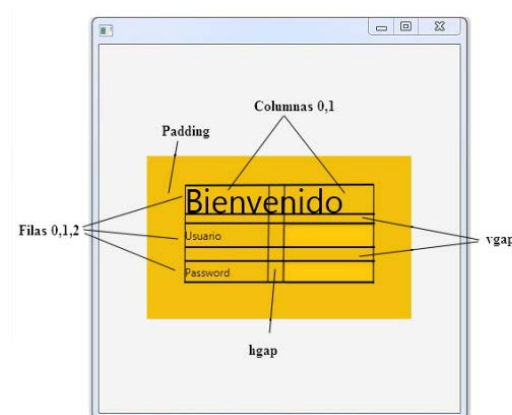


Figura 15

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 la opción de menu Preview puede ver el aspecto de la interfaz). Figura 16

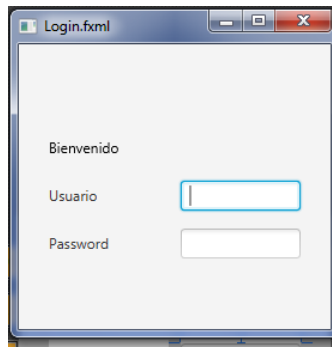


Figura 16

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.

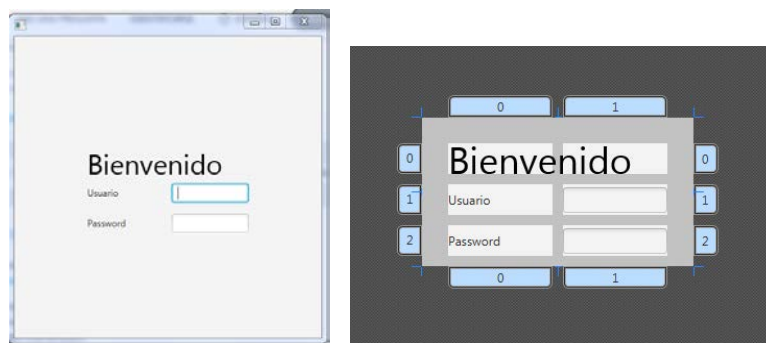


Figura 17

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 un componente Text que se ubicará en una nueva fila de la matriz. Sitúe el componente Text en 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 18



## 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**.

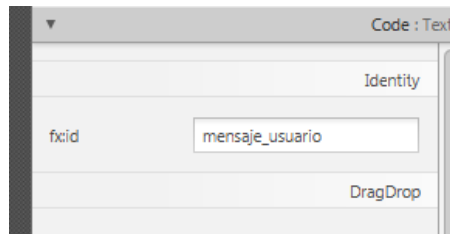


Figura 19. Identificador dl componente Text

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

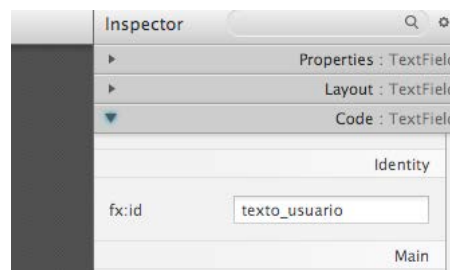


Figura 20. Identificador del TextField

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 21. Asociar un manejador en Scene Builder y en el fichero FXML

figura anterior (21) 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. Actualizar la clase de Java asociada al fichero FXML, clase controlador.

Todo fichero FXML debe de tener asociado un fichero de java con su correspondiente clase. Podemos asociar este fichero de java desde SceneBuilder indicando la clase en el apartado Document> Controller:

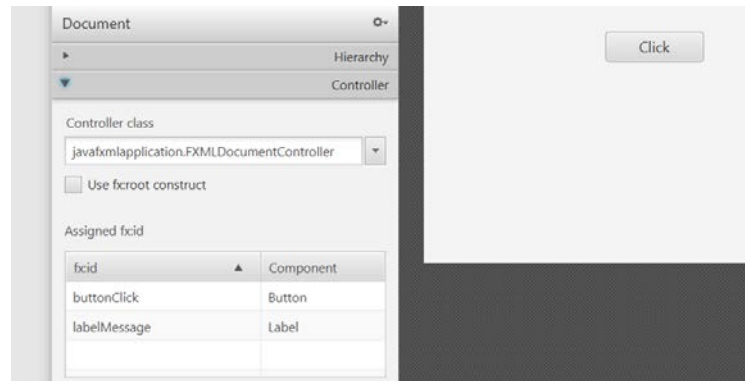


Figura 22. Controller Class

En nuestro proyecto la clase controladora es FXMLDocumentController.java que se encuentra en el paquete javafxmlapplication.

Netbeans tiene una herramienta que permite generar de manera automática y mantener sincronizada esta clase a partir de su definición en el fichero FXML. Esta herramienta hay que utilizarla siempre que modifiquemos el fichero FXML. Para ello seleccionaremos en el explorador del proyecto el fichero FXMLDocument.fxml y con el boton derecho de ratón invocaremos la opcion de menu *Make Controller*

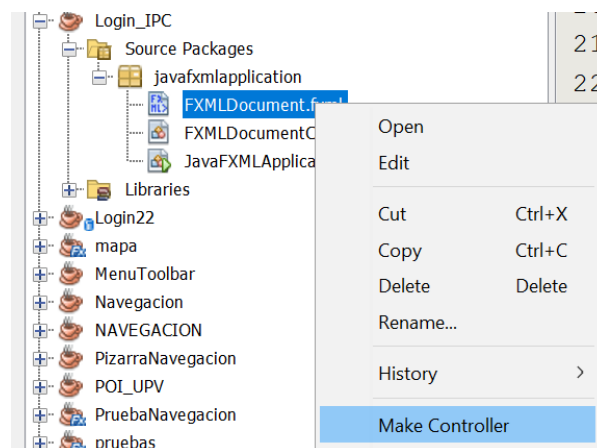


Figura 23. Generación automática de la clase controlador

Se actualiza el fichero FXMLDocumentController.java con los cambios introducidos en el fichero FXML. En la Figura 24 se muestra el resultado.

```

6   package javafxmlapplication;
7
8   import ...9 lines
17
18  /**...4 lines */
22  public class FXMLDocumentController implements Initializable {
23      @FXML
24      private TextField texto_usuario;
25      @FXML
26      private Text mensaje_usuario;
27
28      //=====
29      // you must initialize here all related with the object
30      @Override
31      public void initialize(URL url, ResourceBundle rb) {
32          // TODO
33      }
34      @FXML
35      private void pulsadoIniciar(ActionEvent event) {
36
37      }
38  }
39

```

Figura 24. 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 así 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 25. Código del manejador de eventos

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

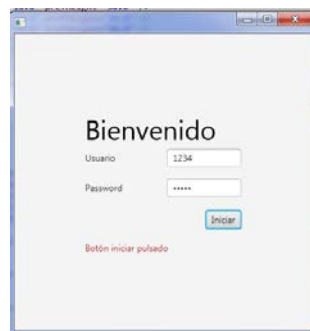


Figura 26 . Resultado Final

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

```

6   package javafxmlapplication;
7
8   import ...5 lines
13
14
15   public class JavaFXMLApplication extends Application {
16
17       @Override
18       public void start(Stage stage) throws Exception {
19           // =====
20           // 1- creación del grafo de escena a partir del fichero FXML
21           FXMLLoader loader= new FXMLLoader(getClass().getResource("FXMLDocument.fxml"));
22           Parent root = loader.load();
23           // =====
24           // 2- creación de la escena con el nodo raíz del grafo de escena
25           Scene scene = new Scene(root, 320,240);
26           // =====
27           // 3- asignación de la escena al Stage que recibe el metodo
28           //     - configuracion del stage
29           //     - se muestra el stage de manera no modal mediante el metodo show()
30           stage.setScene(scene);
31           stage.setTitle("Login");
32           stage.setResizable(false);
33           stage.show();
34       }
35
36       /**...3 lines */
37
38       public static void main(String[] args) {
39           launch(args);
40       }
41
42   }

```

Figura 27. Cambios en la clase principal

## ANEXO I: Crear el nuevo proyecto JavaFX 11 con Netbeans y Ant

Netbeans nos permite crear un proyecto JavaFX utilizando varias herramientas. Podemos crear un proyecto no modular mediante la herramienta Ant o un proyecto modular mediante Maven y Gradle. En este caso vamos a optar por utilizar Ant. Este proceso esta descrito en la información oficial y la podéis encontrar en el siguiente enlace <https://openjfx.io/openjfx-docs/>

### • AÑADIR LA LIBRERIA JAVAFOX11 A NETBEANS

La herramienta 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 los ficheros .jar que dan soporte a javafx. En Netbeans la estrategia consiste en definir una librería y añadir a esta librería todos los ficheros necesarios. Para poder compartir los proyectos entre profesor y alumnos sin hacer modificaciones es necesario tener un mismo nombre para esta librería. En el laboratorio se ha definido la librería **JAVAFX11** y es este es el nombre que debéis definir en vuestra maquina personal.

Antes de definir la librería en Netbeans tenemos que descargar en nuestra maquina el SDK de JavaFX: <https://gluonhq.com/products/javafx/>. Descomprime el fichero zip descargado. Te recomendamos que dejes este sdk en la misma carpeta donde has dejado el jdk, el directorio por defecto es C:\Program Files\Java

El sdk recomendado para Windows y 64 bits y que está en el laboratorio es:

[https://download2.gluonhq.com/openjfx/11.0.2/openjfx-11.0.2\\_windows-x64\\_bin-sdk.zip](https://download2.gluonhq.com/openjfx/11.0.2/openjfx-11.0.2_windows-x64_bin-sdk.zip)

En NetBeans añadiremos nuestra librería en el menú *Tools>Libraries* , pulsaremos *New Library*, y rellenaremos el formulario tal y como aparece en la siguiente figura (26).

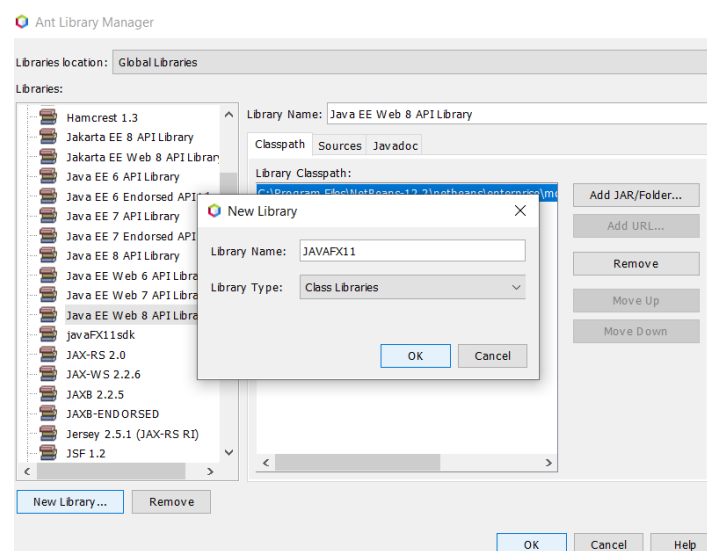


Figura 28. Añadir librería JAVAFX11

Después de crear la librería, tenemos que añadir 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 (figura 29).

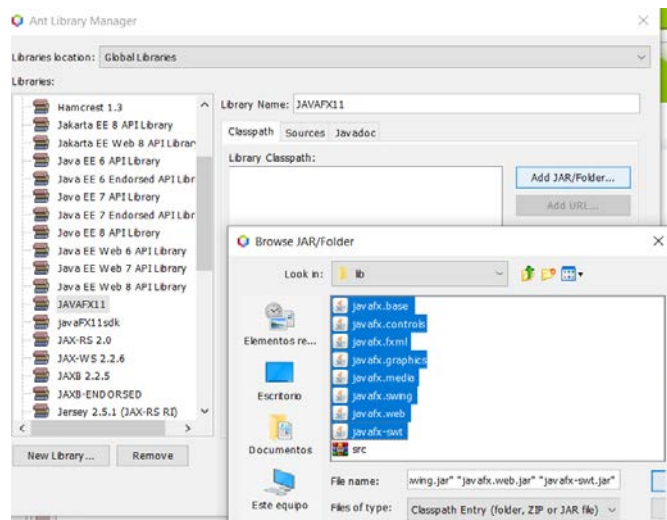


Figura 29. Añadir los jar a la librería

## • Configurar el proyecto de Java

Ya tenemos la librería JAVAFX11 disponible para poder añadirla a un proyecto de Java. Un proyecto de JavaFX con la interfaz definida en un fichero FXML se compone de una clase Aplicación que extiende la clase Application y que implementa el método start(), de un fichero FXML con el diseño de la interfaz y de una clase de java (clase controlador) donde se implementa el código asociado a la funcionalidad de la interfaz. Este proceso de inicialización siempre va a ser el mismo, para agilizarlo os proponemos que utilizéis un proyecto base. Como hemos indicado en el punto 1, podéis descargar el proyecto desde poliformat o desde el repositorio gitub. En la siguiente figura puedes ver el contenido del proyecto:

Como puedes observar en la figura, el proyecto base se llama **IPC\_FXMLCore**.

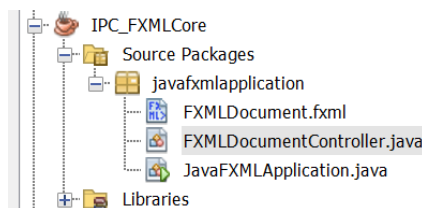


Figura 30. Estructura del proyecto

Las modificaciones que hemos añadido a la configuración del proyecto y que son necesaria para cualquiera de vuestro proyecto son la siguientes:

1. 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**

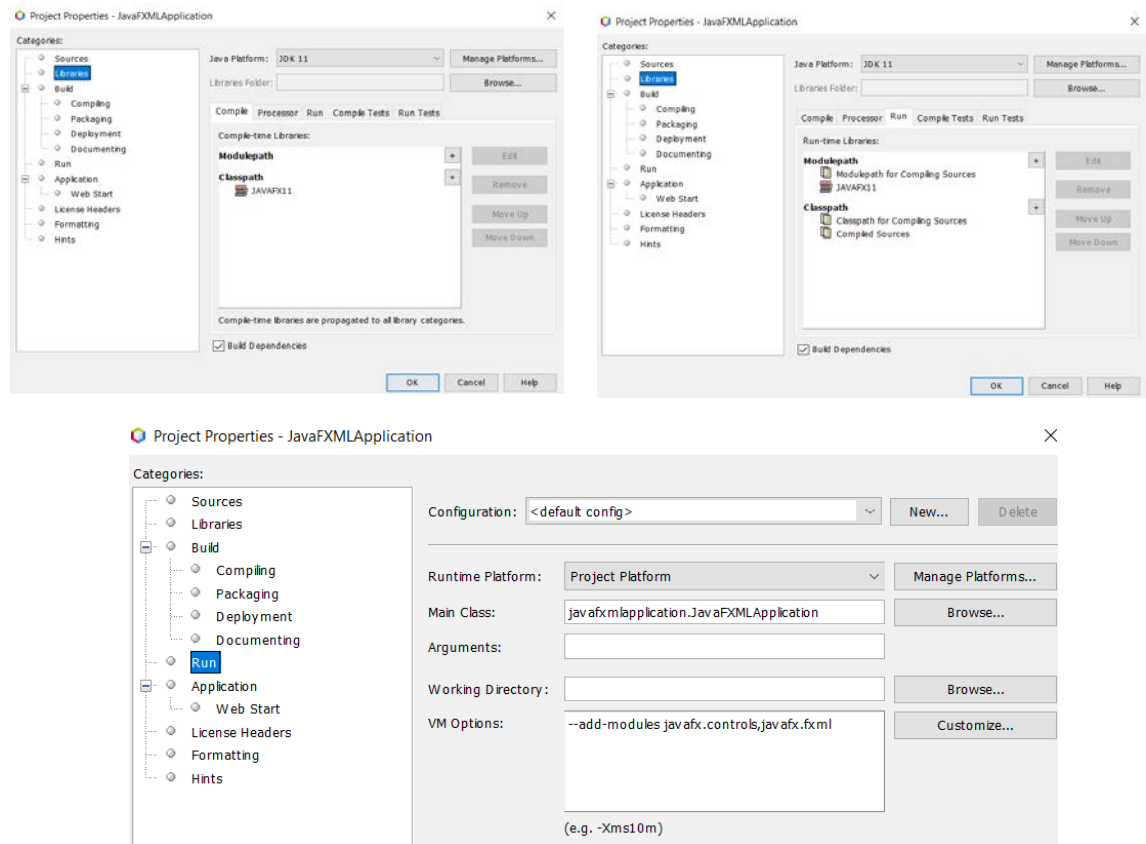


Figura 31. Configuración del proyecto

( si utilizas otros modulos como el webview debes de añadirlos)

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*.

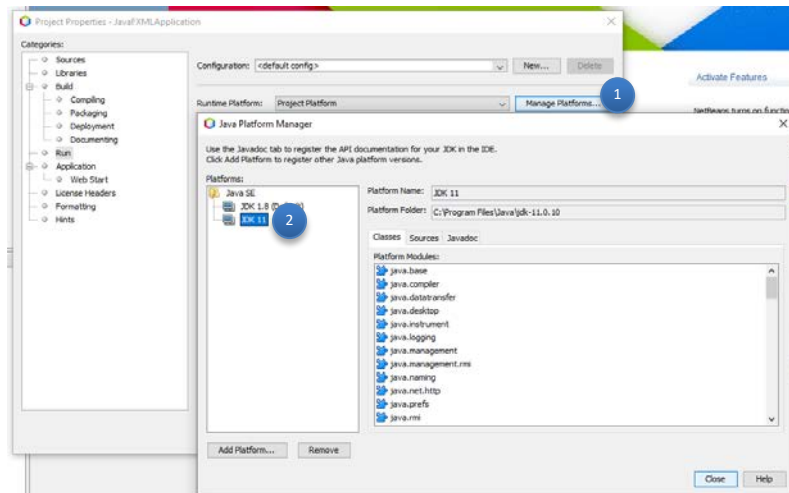


Figura 32. Configuración de la plataforma JDK

Una vez abras el proyecto **JavaFXMLApplication** en Netbeans, procede a cambiar el nombre del proyecto mediante la opción de menú **Rename**

Si quieres cambiar el nombre de los paquetes y/o de los ficheros que contiene el proyecto debes de utilizar la opción **Refactor > Rename**. Esta opción modificará el nombre del fichero y de la clase que contiene y de todos los enlaces a este nombre que se encuentren dentro de los ficheros de código. 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 nombre en la línea 11).

```

8
9
10
11
12
<AnchorPane id="AnchorPane" prefHeight="200" prefWidth="320"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="javafxmlapplication.FXMLDocumentController">
<children>

```

Figura 33. Línea a actualizar en el fichero FXML si cambias el nombre de su clase controladora

También puedes actualizar el nombre de la clase controladora desde SenceBuilder, seleccionando el nuevo nombre en la pestaña Document > **Controller** situada debajo del panel izquierdo, tal y como se muestra en la figura 34. Esta opción es la mas adecuada



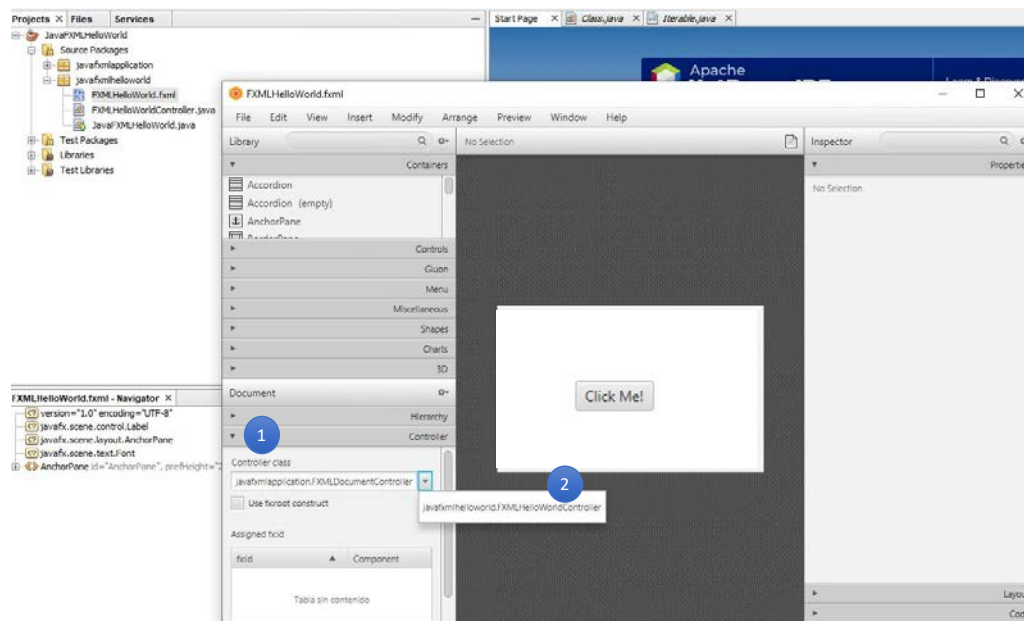


Figura 34 Modificar clase controladora desde SceneBuilder