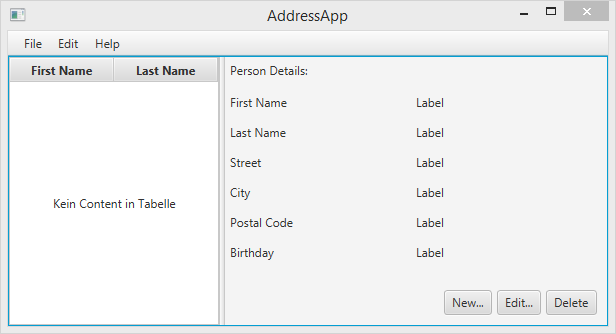
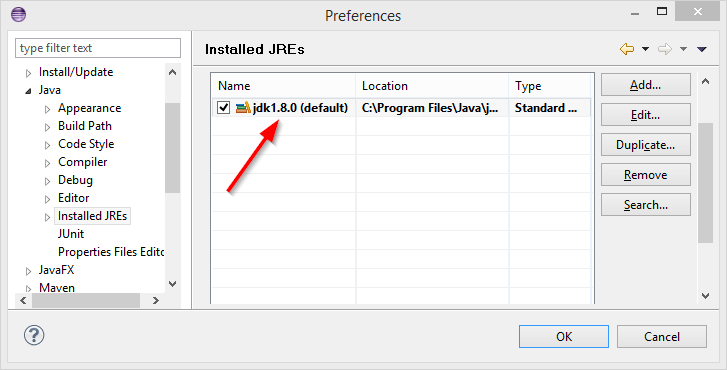
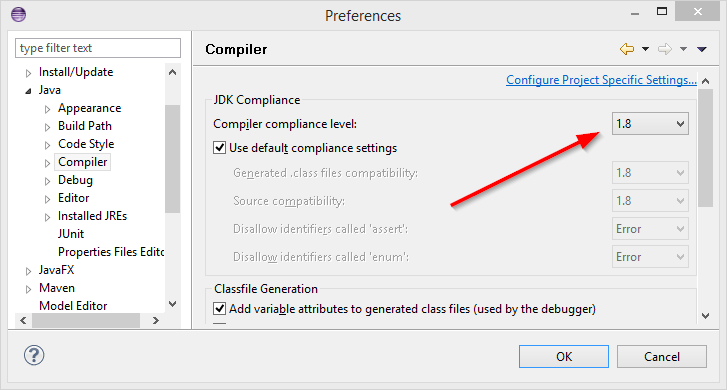
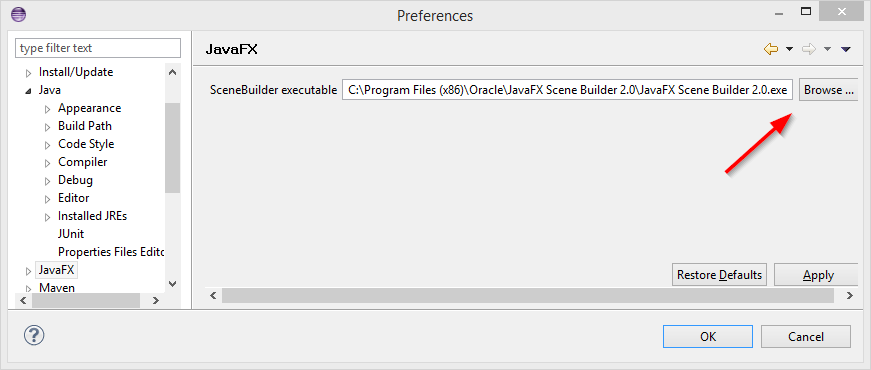
Diseño de aplicaciones:



**Configuración de Eclipse**

Hay que indicarle a Eclipse que use JDK 8 y también dónde se encuentra el ejecutable del Scene Builder:

1. Abre las Preferencias de Eclipse (menú *Window | Preferences* y navega hasta *Java | Installed JREs*.
2. Si no lo tienes el jre1.8 en la lista de JREs, entonces pulsa *Add...*, selecciona *Standard VM* y elige el*Directorio* de instalación (JRE Home directory) de tu JDK 8.
3. Elimina otros JREs o JDKs de tal manera que **JDK 8 se convierta en la opción por defecto**.  
   
4. Navega a *Java | Compiler*. Establece el **nivel de cumplimiento del compilador en 1.8** (Compiler compliance level).  
   
5. Navega hasta *Java | JavaFX*. Especifica la ruta al ejecutable del Scene Builder.  
   

**Enlaces útiles**

Te podría interesar mantener los siguientes enlaces:

* [Java 8 API](http://docs.oracle.com/javase/8/docs/api/) - Documentación (JavaDoc) de las clases estándar de Java
* [JavaFX 8 API](http://docs.oracle.com/javase/8/javafx/api/) - Documentación de las clases JavaFX
* [ControlsFX API](http://controlsfx.bitbucket.org/) - Documentación para el proyecto [ControlsFX](http://fxexperience.com/controlsfx/), el cual ofrece controles JavaFX adicionales
* [Oracle's JavaFX Tutorials](http://docs.oracle.com/javase/8/javafx/get-started-tutorial/get_start_apps.htm) - Tutoriales oficiales de Oracle sobre JavaFX

¡Y ahora, manos a la obra!

**Crea un nuevo proyecto JavaFX**

En Eclipse (con e(fx)clipse instalado) ve a *File | New | Other...* y elige *JavaFX Project*. Especifica el nombre del proyecto (ej. *AddressApp*) y aprieta *Finish*.

Borra el paquete *application* y su contenido si ha sido automáticamente creado.

**Crea los paquetes**

Desde el principio vamos a seguir buenos principios de diseño de software. Algunos de estos principios se traducen en el uso de la arquitectura denominada [**Modelo-Vista-Controlador** (MVC)](http://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador). Esta arquitectura promueve la división de nuestro código en tres apartados claramente definidos, uno por cada elemento de la arquitectura. En Java esta separación se logra mediante la creación de tres paquetes separados.

En el ratón hacemos clic derecho en la carpeta *src*, *New | Package*:

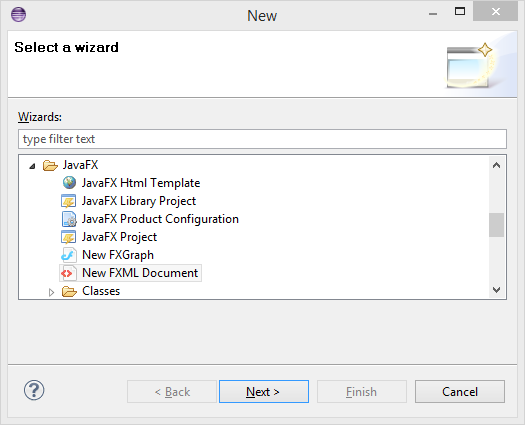
* ch.makery.address - contendrá la *mayoría* de clases de control (C)
* ch.makery.address.model - contendrá las clases del modelo (M)
* ch.makery.address.view - contendrá las vistas (V)

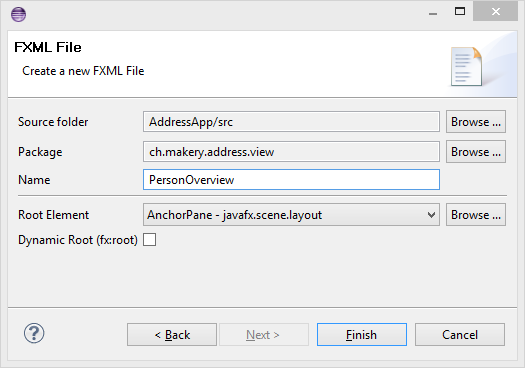
**Nota:** Nuestro paquete dedicado a las vistas contendrá también algunos controladores dedicados exclusivamente a una vista. Les llamaremos **controladores-vista**.

**Crea el archivo FXML de diseño**

Hay dos formas de crear la interfaz de usuario. Programándolo en Java o mediante un archivo XML. Si buscas en Internet encontrarás información relativa a ambos métodos. Aquí usaremos XML (archivo con la extensión .fxml) para casi todo. Encuentro más claro mantener el controlador y la vista separados entre sí. Además, podemos usar la herramienta de edición visual Scene Builder, la cual nos evita tener que trabajar directamente con el XML.

Haz clic derecho el paquete *view* y crea un nuevo archivo FXML (*New | Other | FXML | New FXML Document*) llamado PersonOverview.

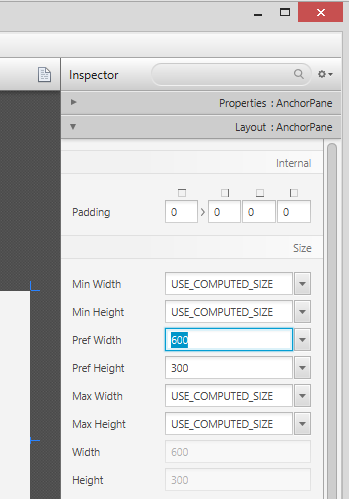
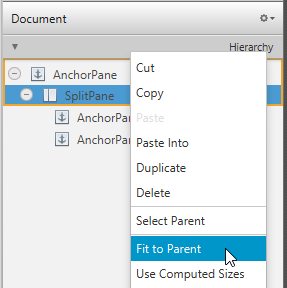
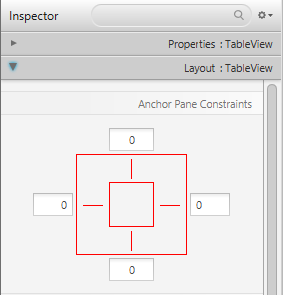
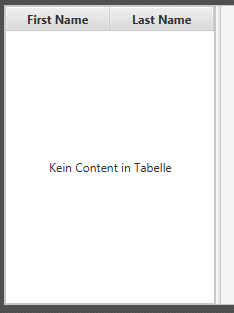
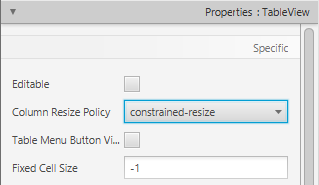
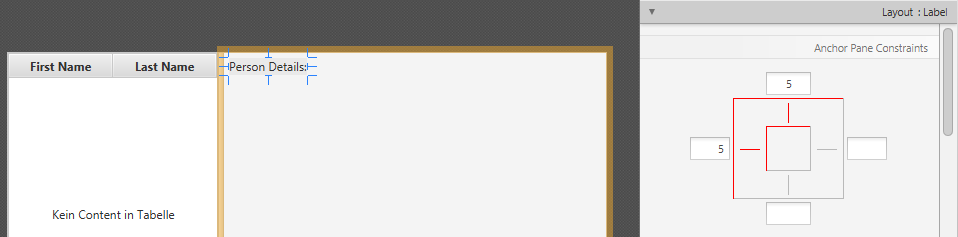
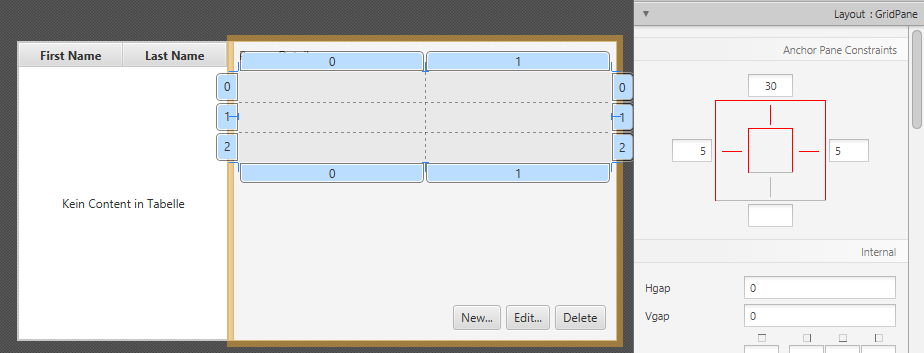
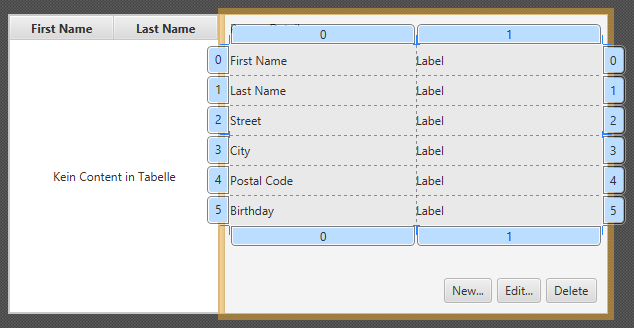




**Diseño mediante Scene Builder**

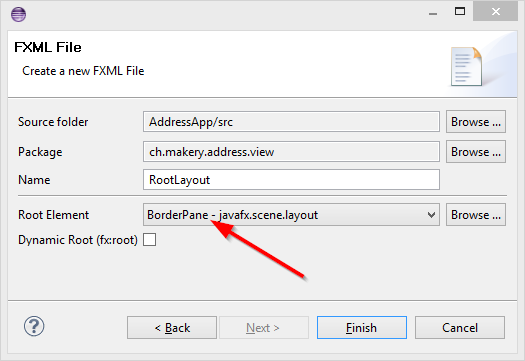
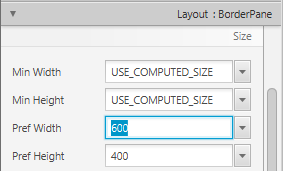
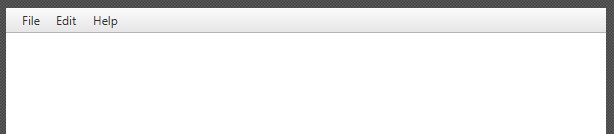
**Nota:** Si no puedes hacerlo funcionar, descarga las fuentes para esta parte del tutorial e inténtalo con el archivo fxml incluido.

Haz clic derecho sobre PersonOverview.fxml y elige *Open with Scene Builder*. Ahora deberías ver el Scene Builder con un *AnchorPane* (visible en la jerarquía de componentes (herramienta Hierarchy) situada a la izquierda).

1. Selecciona el *AnchorPane* en tu jerarquía y ajusta el tamaño en el apartado Layout (a la derecha):  
   
2. Añade un *SplitPane (Horizontal Flow)* arrastrándolo desde la librería (Library) al área principal de edición. Haz clic derecho sobre el *SplitPane* en la jerarquía y elige *Fit to Parent*.  
   
3. Arrastra un *TableView* (bajo *Controls*) al lado izquierdo del *SplitPane*. Selecciona la TableView (no una sola columna) y establece las siguientes restricciones de apariencia (Layout) para la TableView. Dentro de un *AnchorPane* siempre se pueden establecer anclajes (anchors) para los cuatro bordes ([más información sobre Layouts](http://docs.oracle.com/javase/8/javafx/layout-tutorial/builtin_layouts.htm)).  
   
4. Ve al menú *Preview | Show Preview in Window* para comprobar si se visualiza correctamente. Intenta cambiar el tamaño de la ventana. La TableView debería ajustar su tamaño al tamaño de la ventana, pues está "anclada" a sus bordes.
5. Cambia el texto de las columnas (bajo Properties) a "First Name" y "Last Name".  
   
6. Selecciona la *TableView* y elige *constrained-resize* para la *Column Resize Policy* (en Properties). Esto asegura que las columnas usarán siempre todo el espacio que tengan disponible.  
   
7. Añade una *Label* al lado derecho del *SplitPane* con el texto "Person Details" (truco: usa la búsqueda en la librería para encontrar el componente *Label*). Ajusta su apariencia usando anclajes.  
   
8. Añade un GridPane\* al lado derecho, selecciónalo y ajusta su apariencia usando anclajes (superior, derecho e izquierdo).  
   
9. Añade las siguientes etiquetas (Label) a las celdas del *GridPane* .  
   *Nota: Para añadir una fila al GridPane selecciona un número de fila existente (se volverá de color amarillo), haz clic derecho sobre el número de fila y elige "Add Row".*  
   
10. Añade 3 botones a la parte inferior. Truco: Selecciónalos todos, haz click derecho e invoca *Wrap In | HBox*. Esto los pondrá a los 3 juntos en un HBox. Podrías necesitar establecer un espaciado *spacing*dentro del HBox. Después, establece también anclajes (derecho e inferior) para que se mantengan en el lugar correcto.  
    Button Group
11. Ahora deberías ver algo parecido a lo siguiente. Usa el menú *Preview* para comprobar su comportamiento al cambiar el tamaño de la ventana.  
    

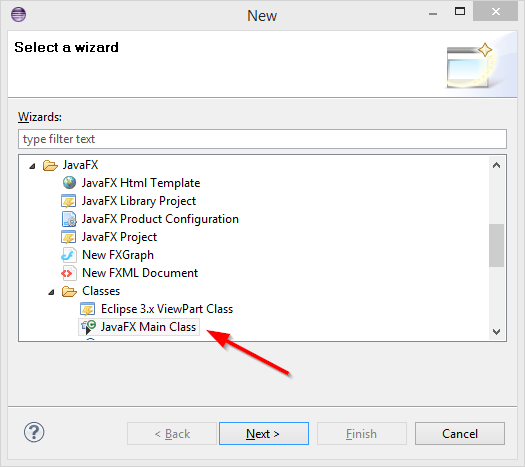
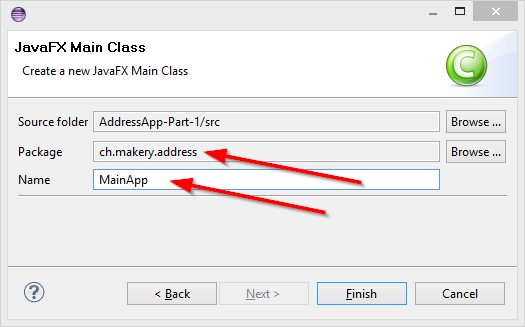
**Crea la aplicación principal**

Necesitamos otro archivo FXML para nuestra vista raíz, la cual contendrá una barra de menús y encapsulará la vista recién creada PersonOverview.fxml.

1. Crea otro archivo FXML dentro del paquete view llamado RootLayout.fxml. Esta vez, elige *BorderPane*como elemento raíz..  
   
2. Abre RootLayout.fxml en el Scene Builder.
3. Cambia el tamaño del *BorderPane* con la propiedad *Pref Width* establecida en 600 y *Pref Height* en 400.  
   
4. Añade una *MenuBar* en la ranura superior del *BorderPane*. De momento no vamos a implementar la funcionalidad del menú.  
   

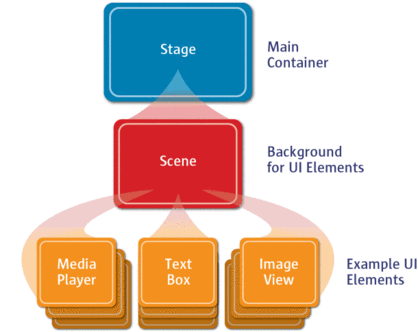
**La clase principal en JavaFX**

Ahroa necesitamos crear la **clase java principal**, la cual iniciará nuestra aplicación medianteRootLayout.fxml y añadirá la vista PersonOverview.fxml en el centro.

1. Haz clic derecho en el proyecto y elige *New | Other | JavaFX | classes | JavaFX Main Class*.  
   
2. Llama a esta clase MainApp y ponla en el paquete de controladores ch.makery.address (nota: este es el paquete padre de los paquetes view y model).  
   

La clase generada (MainApp.java) extiende a la clase Application y contiene dos métodos. Esta es la estructura básica que necesitamos para ejecutar una Aplicación JavaFX. La parte más importante para nosotros es el método start(Stage primaryStage). Este método es invocado automáticamente cuando la aplicación es lanzada desde el método main.

Como puedes ver, el método start(...) tomo un Stage como parámetro. El gráfico siguiente muestra la estructura de cualquier aplicación JavaFX:

  
*Fuente de la imagen:*[*http://www.oracle.com*](http://www.oracle.com/)

**Es como una obra de teatro**: El Stage (escenario) es el contenedor principal, normalmente una ventana con borde y los típicos botones para maximizar, minimizar o cerrar la ventana. Dentro del Stage se puede añadir una Scene (escena), la cual puede cambiarse dinámicamente por otra Scene. Dentro de un Scene se añaden los nodos JavaFX, tales como AnchorPane, TextBox, etc.

Para tener más información puedes consultar [Working with the JavaFX Scene Graph](http://docs.oracle.com/javase/8/javafx/scene-graph-tutorial/scenegraph.htm).

Abre el archivo MainApp.java y reemplaza todo su código con el código siguiente:

package ch.makery.address;

import java.io.IOException;

import javafx.application.Application;

import javafx.fxml.FXMLLoader;

import javafx.scene.Scene;

import javafx.scene.layout.AnchorPane;

import javafx.scene.layout.BorderPane;

import javafx.stage.Stage;

public class MainApp extends Application {

private Stage primaryStage;

private BorderPane rootLayout;

@Override

public void start(Stage primaryStage) {

this.primaryStage = primaryStage;

this.primaryStage.setTitle("AddressApp");

initRootLayout();

showPersonOverview();

}

/\*\*

\* Initializes the root layout.

\*/

public void initRootLayout() {

try {

// Load root layout from fxml file.

FXMLLoader loader = new FXMLLoader();

loader.setLocation(MainApp.class.getResource("view/RootLayout.fxml"));

rootLayout = (BorderPane) loader.load();

// Show the scene containing the root layout.

Scene scene = new Scene(rootLayout);

primaryStage.setScene(scene);

primaryStage.show();

} catch (IOException e) {

e.printStackTrace();

}

}

/\*\*

\* Shows the person overview inside the root layout.

\*/

public void showPersonOverview() {

try {

// Load person overview.

FXMLLoader loader = new FXMLLoader();

loader.setLocation(MainApp.class.getResource("view/PersonOverview.fxml"));

AnchorPane personOverview = (AnchorPane) loader.load();

// Set person overview into the center of root layout.

rootLayout.setCenter(personOverview);

} catch (IOException e) {

e.printStackTrace();

}

}

/\*\*

\* Returns the main stage.

\* @return

\*/

public Stage getPrimaryStage() {

return primaryStage;

}

public static void main(String[] args) {

launch(args);

}

}