Calculadora en JavaFX

1. Introducción

Vamos a intentar juntar todo lo aprendido en una guía para realizar una aplicación `JavaFX` con `SceneBuilder` i `NetBeans`, siguiendo el modelo `MVC`.

Necesitaras:

- Apache Netbeans 16 o posterior
- OpenJDK 17 (seguramente funcionará con una posterior)
- JavaFx 19
- SceneBuilder

2. Crear proyecto

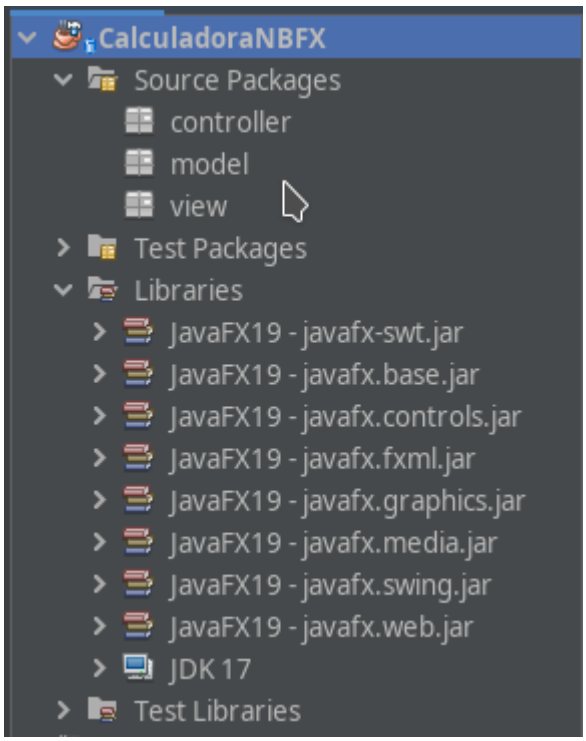
Vamos a crear el proyecto como siempre `Java with Ant` no necesitamos el asistente de `JavaFX`.

En nuestro caso llamamos al proyecto `CalculadoraNBFX` y desmarcamos la opción de que nos cree una clase `main`.

Ahora para seguir el modelo **MVC** crearemos los tres *packages* (`controller`, `model` y `view`).

Ahora añadimos al proyecto la librería de `JavaFX19` que hemos creado en otros documentos de la unidad.

En este momento nuestro proyecto debería tener este aspecto:



Desactivamos la opción de compilar al guardar del proyecto y añadimos las propiedades correctas a las `VM options` del proyecto. Tal y como hemos visto en otros documentos de la unidad.

3. Modelo

Para la calculadora necesitaremos un modelo que se encargue de realizar las distintas operaciones de nuestra calculadora. Para ello crearemos un nuevo fichero `Operaciones.java` dentro del paquete `model` con el siguiente contenido:

```java
package model;

public class Operaciones {
    private double operador1;
    private double operador2;

    public Operaciones(double operador1, double operador2) {
        this.operador1 = operador1;
        this.operador2 = operador2;
    }

    public double getOperador1() {
        return operador1;
    }

    public void setOperador1(double operador1) {
        this.operador1 = operador1;
    }

    public double getOperador2() {
        return operador2;
    }

    public void setOperador2(double operador2) {
        this.operador2 = operador2;
    }

    public double suma(){
        return this.operador1+this.operador2;
    }
    public double resta(){
        return this.operador1-this.operador2;
    }
    public double multiplicacion(){
        return this.operador1*this.operador2;
    }
    public double division(){
        return this.operador1/this.operador2;
    }
}
```

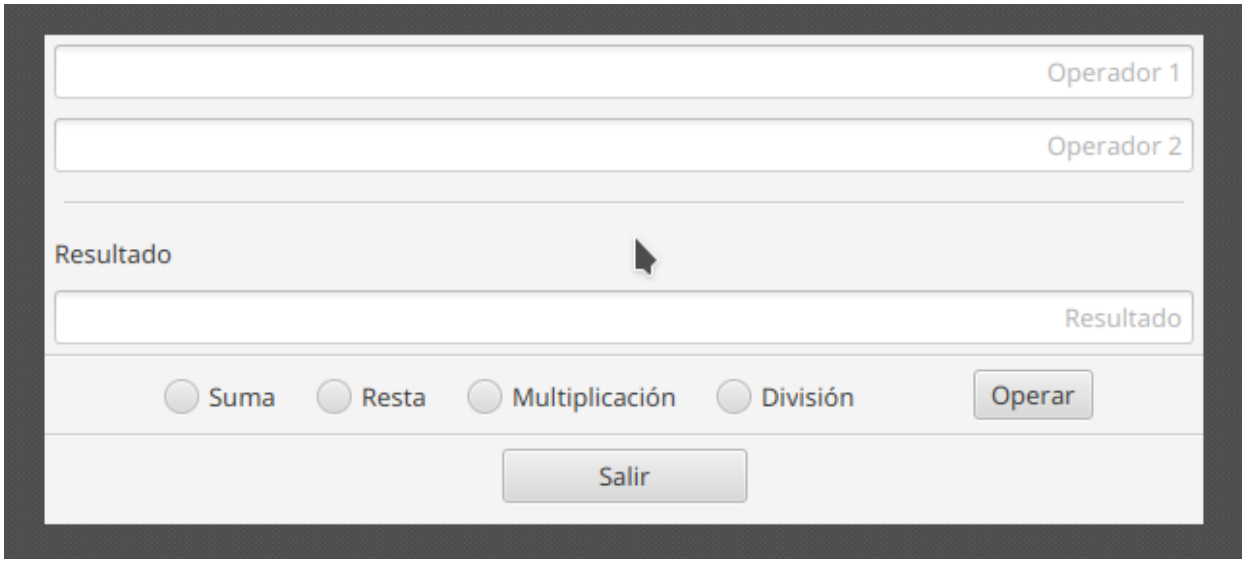
Fíjate que este es un modelo muy simple, con dos atributos, un constructor, sus *getters* y *setters* y las cuatro operaciones básicas de nuestra calculadora (sumar, restar, multiplicar y dividir).

4. Vista

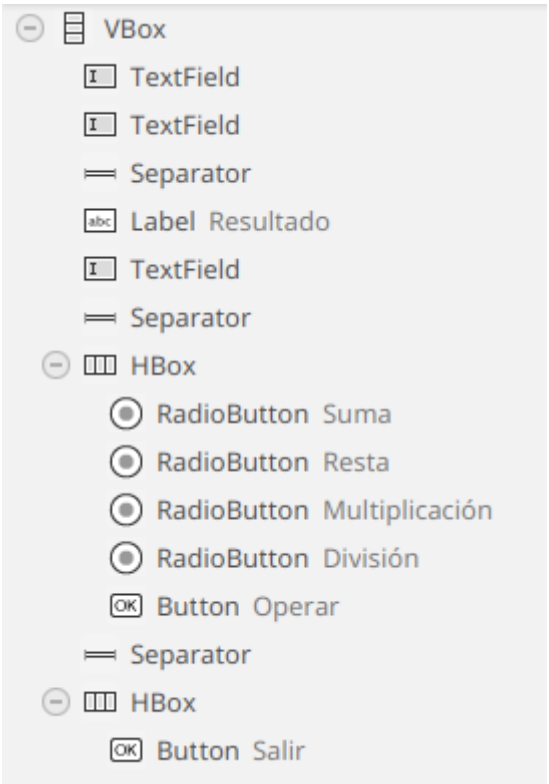
Ahora vamos añadir un nuevo fichero `Empty FXML` al paquete `view`. Llamaremos Calculadora al nuevo fichero, elegiremos el paquete correspondiente y pulsaremos finalizar (no crearemos el controlador ni la hoja de estilos, esto lo haremos manualmente más adelante).

Si tenemos correctamente configurado el **SceneBuilder**, tal y como hemos visto en otros documentos, al hacer doble clic sobre el fichero `Calculadora.fxml` debería abrirse con él.

Ahora deberías crear una ventana similar a esta:



Este ejemplo tiene la siguiente jerarquía:



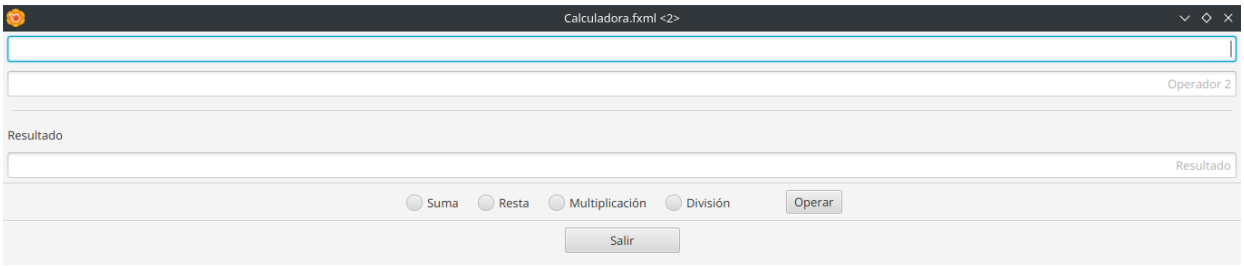
Recuerda dar nombre a todos los componentes en la pestaña `code` al campo `fx:id`.

`txtOperador1`, `txtOperador2` y `txtResultado` para los `TextField's`

`rbSuma`, `rbResta`, `rbMultiplicación`, `rbDivision` para los `RadioButton's`

Desactiva el `txtResultado`, para que no sea editable.

Crea los contenedores y ajusta sus alineaciones, así como los márgenes y espaciadores de los elementos que contienen, de manera que si amplias la ventana al máximo quede algo similar a esto:



También debes añadir la acciones `ON ACTION` dentro de la pestaña `code` para los botones:

`btnSalir`: # `salir`

`btnOperar`: # `operar`

5. Controladores

Necesitaremos dos archivos dentro del package de `controller`:

1. El controlador para la vista de la calculadora.
2. La clase main que cargará la vista principal

5.1. `CalculadoraController.java`

Realizar el controlador para la vista es muy sencillo y automático. Debemos hacer clic derecho sobre el archivo `Calculadora.fxml` y elegir la opción `Make Controller` de **NetBeans**. Esta opción creará el controlador en el mismo paquete de `view`, y posteriormente deberemos moverlo a su correspondiente lugar dentro del paquete `controller`.

Recuerda también cambiarlo en el código del archivo `Calculadora.fxml` (botón derecho `edit`) de:

```
<VBox maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="253.0" prefWidth="600.0" xmlns="http://javafx.com/javafx/19" xmlns:fx="http://javafx.com/fxml/1" fx:controller="view.CalculadoraController">
```

a:

```
<VBox maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="253.0" prefWidth="600.0" xmlns="http://javafx.com/javafx/19" xmlns:fx="http://javafx.com/fxml/1" fx:controller="controller.CalculadoraController">
```

Ahora, dentro del `CalculadoraController.java` agregaremos el código necesario para gestionar las acciones de los botones, y además asegurarnos que los *radio buttons* son auto-excluyentes:

Acción `salir`:

```
    @FXML
    private void salir(ActionEvent event) {
        Stage stage = (Stage) btnSalir.getScene().getWindow();
        stage.close();
    }
```

Acción `operar`:

```
    @FXML
    private void operar(ActionEvent event) {
        try {
            double op1 = Double.parseDouble(this.txtOperador1.getText());
            double op2 = Double.parseDouble(this.txtOperador2.getText());
            Operaciones op = new Operaciones(op1, op2);
            if (this.rbSuma.isSelected()) {
                this.txtResultado.setText(String.valueOf(op.suma()));
            } else if (this.rbResta.isSelected()) {
```

```
                this.txtResultado.setText(String.valueOf(op.resta()));
            } else if (this.rbMultiplicacion.isSelected()) {
                this.txtResultado.setText(String.valueOf(op.multiplicacion()));
            } else if (this.rbDivision.isSelected()) {
                if (op2 != 0) {
                    this.txtResultado.setText(String.valueOf(op.division()));
                } else {
                    Alert alert = new Alert(Alert.AlertType.ERROR);
                    alert.setHeaderText(null);
                    alert.setTitle("Error");
                    alert.setContentText("El operador 2 no puede ser 0.");
                    alert.showAndWait();
                }
            }
        } catch (NumberFormatException numberFormatException) {
            Alert alert = new Alert(Alert.AlertType.ERROR);
            alert.setHeaderText(null);
            alert.setTitle("Error");
            alert.setContentText("Formato incorrecto de algun operando");
            alert.showAndWait();
        }
    }
```

Recuerda realizar el *import* del `model.Operaciones`:

```
import model.Operaciones;
```

Acción `initialize`:

```
    @Override
    public void initialize(URL url, ResourceBundle rb) {
        ToggleGroup tgRadio = new ToggleGroup();
        rbSuma.setToggleGroup(tgRadio);
        rbMultiplicacion.setToggleGroup(tgRadio);
        rbResta.setToggleGroup(tgRadio);
        rbDivision.setToggleGroup(tgRadio);
    }
```

El método `initialize` será llamado al instanciar el controlador y generará un `ToggleGroup` de manera que solo podamos seleccionar una de las cuatro opciones disponibles.

5.2. `main.java`

Por último solo nos queda añadir la clase `main`, que contendrá el método `main` que lanzará la aplicación JavaFX.

Para ello en el paquete controller pulsamos botón derecho y añadimos un fichero de tipo `JavaFX Main Class` y le llamaremos `Main.java`.

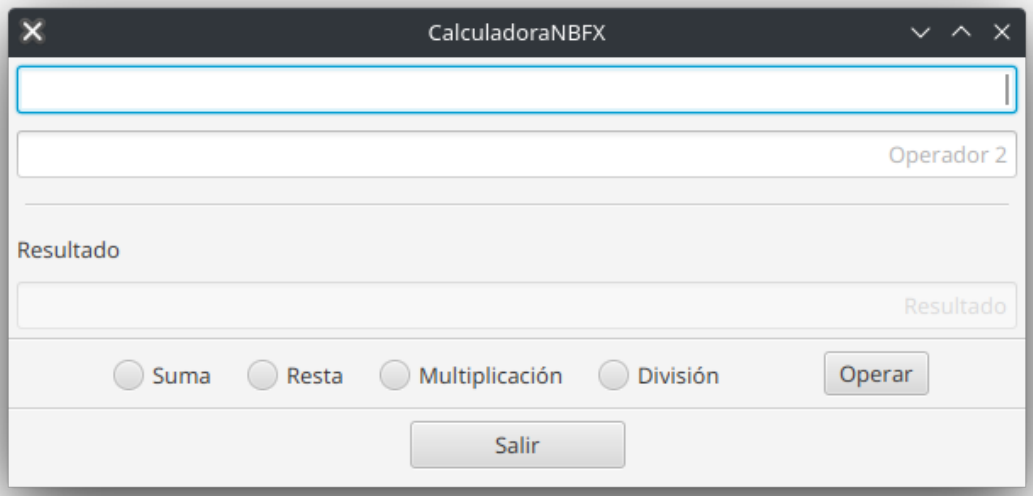
Netbeans genera un método start de ejemplo, que nostotros sustituiremos el siguiente código para que cargue nuestra vista:

```
    @Override
    public void start(Stage primaryStage) {
        try {
            Parent root =
FXMLLoader.load(getClass().getResource("../view/Calculadora.fxml"));
            Scene scene = new Scene(root);
            primaryStage.setTitle("CalculadoraNBFX");
            primaryStage.setScene(scene);
            primaryStage.show();
        } catch (IOException e) {
            System.out.println(e.getMessage());
        }
    }
```

6. Primer lanzamiento

La primera vez que ejecutemos el proyecto nos pedirá asignar la clase `main` que contiene el método `main`, debemos elegir la clase `model.Main`.

Si todo ha ido bien debería aparecer nuestra calculadora en pantalla:



7. Píldoras informáticas relacionadas

- https://www.youtube.com/playlist?list=PLNjWMbvTJAIjLRW2qyuc4DEgFVW5YFRSR
- https://www.youtube.com/playlist?list=PLaxZkGlLWHGUWZxuadN3J7KKaICRlhz5-

8. Fuentes de información

- Apuntes de Jose Antonio Diaz-Alejo
- https://github.com/openjfx/openjfx-docsopen
- https://github.com/openjfx/samples
- FXDocs
- https://openjfx.io/openjfx-docs/
- https://docs.oracle.com/javase/8/javafx/user-interface-tutorial
- https://github.com/JonathanGiles/scenic-view