

4. Gestión de eventos

4.1. Tipos de eventos

Los principales tipos de eventos que se pueden gestionar con JavaFX son:

ActionEvent: un evento que representa algún tipo de acción. Se trata de un evento genérico cuya utilidad concreta debe especificarse más tarde. Este tipo de evento se usa ampliamente para representar una variedad de acciones.

InputEvent: un evento que indica una entrada de usuario, como cuándo se ha pulsado un botón, cuándo se ha pulsado una tecla, si se ha liberado la tecla pulsada y otros usos similares.

WindowEvent: evento relacionado con acciones de mostrar/ocultar ventanas.

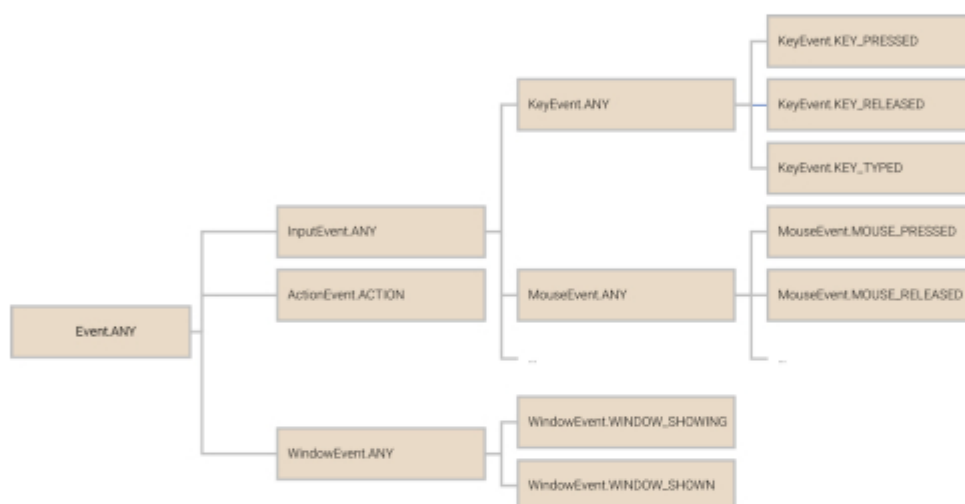


Fig 13.14 Tipos de eventos en JavaFX

4.2. Asignar el controlador a la vista

Para poder manejar los eventos de usuario, lo primero que debemos hacer es asignar a la vista un controlador.

Para ello, debemos crear una clase Java para el controlador.



IMPORTANTE

El nombre del controlador debería ser el nombre de la vista sin la extensión `.fxml`, más la palabra *Controller*. De esta forma, no tendremos ningún problema a la hora de identificar qué controlador corresponde a cada una de las vistas.

Por ejemplo:

Para la vista ***Ventana.fxml***, crearíamos el controlador ***VentanaController.java***.

Por otra parte, existe el método *initialize()*, que podemos utilizar para inicializar ciertos componentes cuando se abre la ventana. Es como el constructor de la clase, pero se diferencia de este en que, desde este método, sí que se puede acceder a los componentes gráficos antes de que se abra la ventana. La estructura de la clase, provista de una ventana con un solo botón de aceptar, quedaría de la siguiente manera:

```
import javafx.fxml.FXML;
import javafx.scene.control.Button;

public class VentanaController {

    // Declaración de componentes gráficos
    @FXML private Button boton_Aceptar;

    //Método que se ejecuta al crear la ventana
    @FXML
    void initialize() {

    }

    //Otros métodos y eventos
}
```

En **Ventana.fxml** deberemos indicar el controlador que se va a utilizar mediante el atributo **fx:controller** ubicado en el contenedor principal de la ventana.

```
<AnchorPane prefWidth="600.0" prefHeight="400.0"
xmlns="http://javafx.com/javafx/11.0.1" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="controlador.VentanaController">
```

4.3. Asignar eventos a los componentes gráficos

Opción A.- Asignar evento en la vista (.fxml)

Desde SceneBuilder configuramos lo siguiente:

- **fxid** donde indicaremos el nombre del componente que utilizaremos en el controlador.
- **On Action**: evento que se producirá cuando se realice cualquier interacción con el botón.
- **OnMouseClicked**: es el evento específico que podemos utilizar cuando se haga clic en el botón.

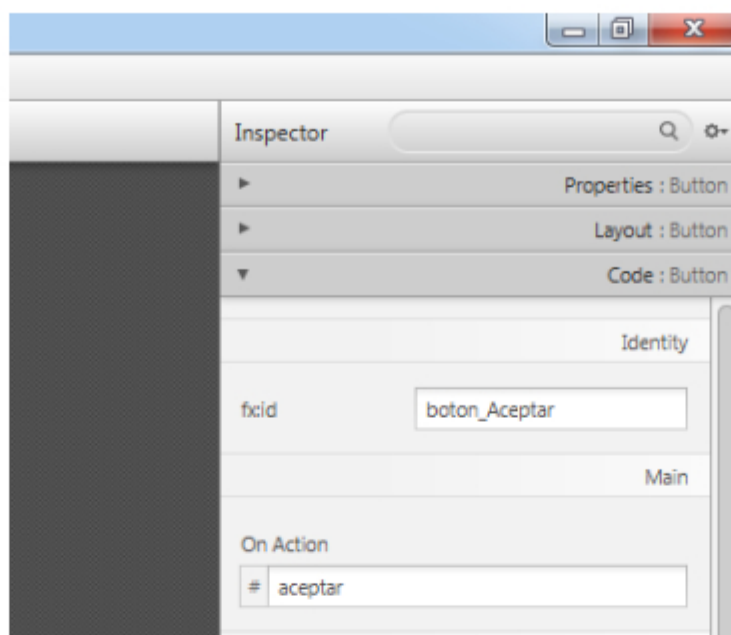


Fig 13.15 Vista de SceneBuilder para asignar eventos

El resultado en **Ventana.fxml** quedará de la siguiente manera:

```
<Button fx:id="boton_Aceptar" layoutX="120.0" layoutY="180.0"
mnemonicParsing="false" onMouseClicked="#aceptar" text="ACEPTAR"/>
```

Además, en **VentanaController.java**, crearemos el método que realizará las acciones correspondientes cuando el usuario pulse el botón ACEPTAR.

```
public class VentanaController {

    // Atributos graficos FXML
    @FXML private Button boton_Aceptar;

    @FXML
    void aceptar(ActionEvent event) {

        // Acciones al pulsar el botón de ACEPTAR

    }

}
```

Opción B.- Asignar eventos, desde el controlador, al inicializar la vista.

En este caso no añadimos ningún atributo en la vista, **Ventana.fxml**.

```
<Button fx:id="boton_Aceptar" layoutX="120.0" layoutY="180.0"
mnemonicParsing="false" text="ACEPTAR"/>
```

En el controlador **VentanaController.java** se deben realizar dos acciones:

- En el método *initialize()*, asignar el evento al botón ACEPTAR.
- Crear el método *aceptar()*, donde se programarán las acciones que se van a realizar cuando se pulse el botón.

```
public class VentanaController {  
  
    // Atributos graficos FXML  
    @FXML private Button boton_Aceptar;  
  
    @Override  
    public void initialize() {  
  
        // Asignar evento al botón ACEPTAR  
        boton_Aceptar.setOnMouseClicked((event) -> aceptar ());  
    }  
  
    // Métodos y eventos  
    private void aceptar() {  
  
        // Acciones al pulsar el botón de ACEPTAR  
  
    }  
}
```

