Java FX

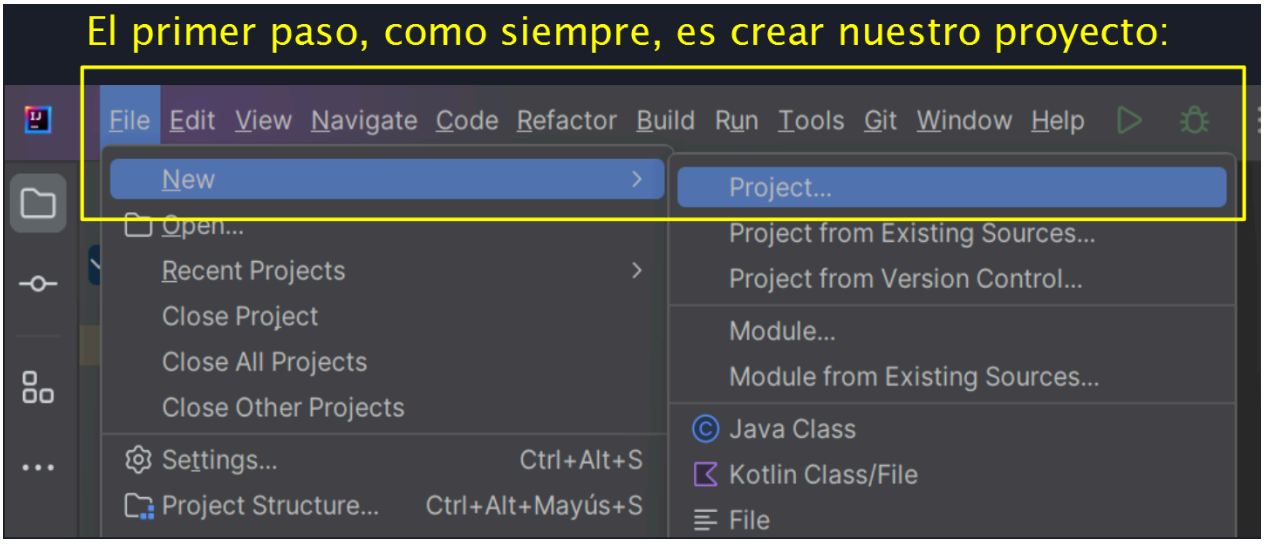
Es un conjunto de bibliotecas y herramientas de desarrollo que permite crear aplicaciones de escritorio con interfaces gráficas de usuario (gui, por sus siglas en inglés) en java.

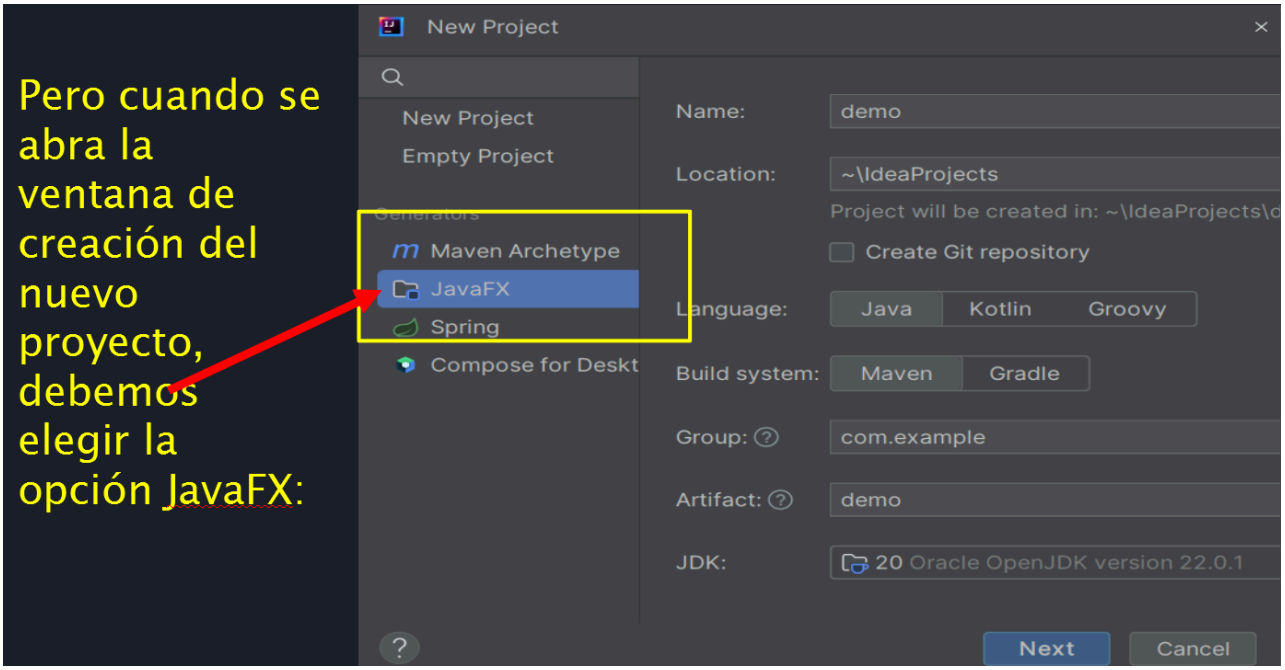
Proporciona una amplia gama de controles y componentes gráficos para diseñar interfaces intuitivas y atractivas, así como funcionalidades para manejar eventos, animaciones y multimedia.
Es una alternativa a la biblioteca SWING, pero con mayor capacidad de personalización y mejores características visuales.

Ventajas:

- Integración nativa con Java
- Diseño de interfaces gráficas atractivas.
- Soporte para multimedia.
- Escalabilidad.
- Compatibilidad multiplataforma.

Manual de uso:





Clase Application

- ➔ Es la base de cualquier aplicación JavaFX.
- ➔ Debe ser extendida por cualquier clase principal de la aplicación.
- ➔ Proporciona el método start() que se ejecuta al iniciar la aplicación.

```java
public class MiAplicacion extends Application {

    @Override
    public void start(Stage primaryStage) {
        // Configurar la interfaz gráfica
        Button button = new Button("Haz clic");
        Scene scene = new Scene(button, 200, 100);

        // Configurar la ventana principal
        primaryStage.setTitle("Mi Aplicación");
        primaryStage.setScene(scene);

        // Mostrar la ventana principal
        primaryStage.show();
    }

    public static void main(String[] args) {
        launch(args);
    }
```

Clase Stage

- ➔ Es la ventana principal de la aplicación.
- ➔ Puede contener una o varias escenas.

```java
public class MiAplicacion extends Application {

    @Override
    public void start(Stage primaryStage) {
        // Configurar la interfaz gráfica
        Button button = new Button("Haz clic");
        Scene scene = new Scene(button, 200, 100);

        // Configurar la ventana principal
        primaryStage.setTitle("Mi Aplicación");
        primaryStage.setScene(scene);

        // Mostrar la ventana principal
        primaryStage.show();
    }

    public static void main(String[] args) {
        launch(args);
    }
}
```

Métodos importantes de la clase Stage:

Ventanas:

- ➔ setTitle(“Nombre App”)
- ➔ getIcons()

Dimensiones:

- ➔ setWidth()
- ➔ setHeight()
- ➔ getWidth()
- ➔ getHeight()

Escenas:

- ➔ setScene(Scene)

Estado y visibilidad:

- ➔ show()
- ➔ setMaximized()
- ➔ hide()
- ➔ setIconified()

Clase Scene

Es el contenedor principal de los elementos de la interfaz gráfica.

- Nodos (Node): La clase Scene puede contener nodos como botones, etiquetas, campos de texto, etc.
- Contenedor Raíz: La escena contiene un contenedor raíz, como un Pane o un Group, que actúa como el nodo principal para organizar y colocar otros nodos gráficos en la escena.
- Dimensiones: Puedes setear u obtener el ancho y el alto de la escena utilizando los métodos:
  - ➔ setWidth()
  - ➔ setHeight()
  - ➔ getWidth()
  - ➔ getHeight()

```
public class MiAplicacion extends Application {

    @Override
    public void start(Stage primaryStage) {
        // Configurar la interfaz gráfica
        Button button = new Button("Haz clic");
        Scene scene = new Scene(button, 200, 100);

        // Configurar la ventana principal
        primaryStage.setTitle("Mi Aplicación");
        primaryStage.setScene(scene);

        // Mostrar la ventana principal
        primaryStage.show();
    }

    public static void main(String[] args) {
        launch(args);
    }
}
```

- Fondo: Se puede establecer:
  - Un fondo para la escena utilizando el método setFill().
  - Un color sólido (Color).
  - Un gradiente (LinearGradient o RadialGradient).

Puedes registrar controladores de eventos en la escena para responder a eventos de interacción del usuario, como clics de mouse, pulsaciones de teclas, movimiento del mouse, entre otros.
Los métodos setOn...() te permiten asociar un controlador de eventos específico con la escena.

setOnAction() : Registra un controlador de eventos para el evento de acción, que generalmente se activa cuando se hace clic en un botón u otro control interactivo.

```
Button button = new Button("Click Me");
button.setOnAction(event -> {
    System.out.println("Button clicked!");
});
```

setOnMouseClicked() :Registra un controlador de eventos para el evento de clic del mouse.

```
Node node = new Rectangle(100, 100);
node.setOnMouseClicked(event -> {
    System.out.println("Mouse clicked!");
});
```

setOnKeyPressed(): Registra un controlador de eventos
para el evento de pulsación de tecla.

```
TextField textField = new TextField();
textField.setOnKeyPressed(event -> {
    System.out.println("Key pressed: " + event.getCode());
});
```

setOnMouseEntered() :Registra un controlador de eventos para el evento de entrada del mouse a un nodo.

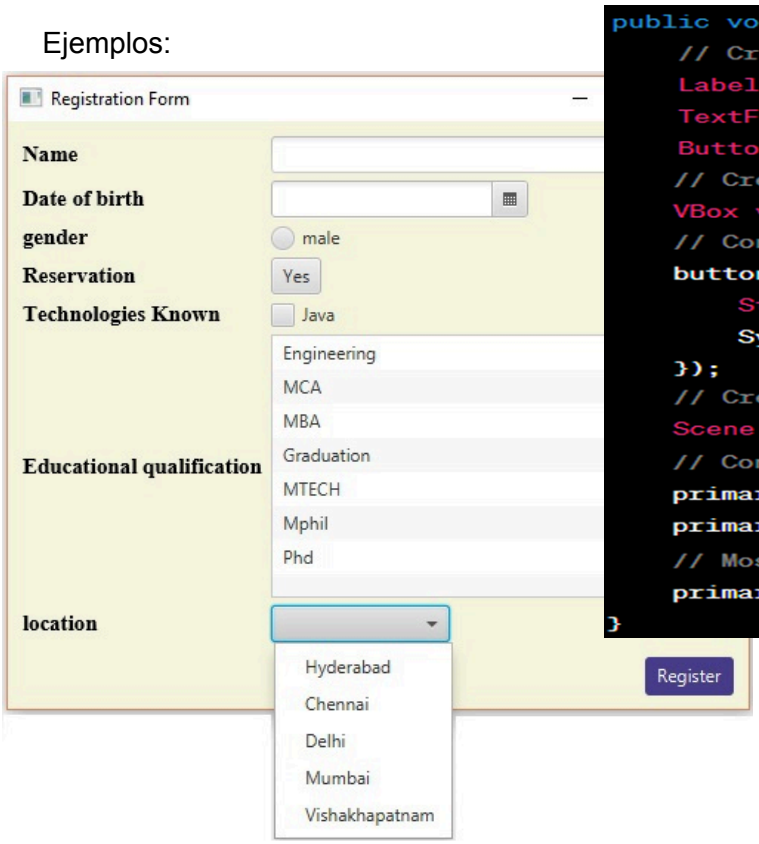
```
Node node = new Circle(50);
node.setOnMouseEntered(event -> {
    System.out.println("Mouse entered the node!");
});
```

Controls

JavaFX proporciona una variedad de controles predefinidos que se pueden usar en la interfaz gráfica. Algunos ejemplos son:

- Button
- Label
- TextField
- PasswordField:
- TextArea
- ComboBox
- heckBox
- RadioButton
- ToggleButton
- FileChooser
- ProgressIndicator

Ejemplos:




```java
public void start(Stage primaryStage) {
    // Crear los controles
    Label label = new Label("Ingrese su nombre:");
    TextField textField = new TextField();
    Button button = new Button("Saludar");
    // Crear el VBox y agregar los controles
    VBox vbox = new VBox(label, textField, button);
    // Configurar el evento del botón
    button.setOnAction(e -> {
        String nombre = textField.getText();
        System.out.println("¡Hola, " + nombre + "!");
    });
    // Crear la escena
    Scene scene = new Scene(vbox, 300, 150);
    // Configurar la ventana principal
    primaryStage.setTitle("Ejemplo de Controles");
    primaryStage.setScene(scene);
    // Mostrar la ventana principal
    primaryStage.show();
}
```

Layouts

JavaFX proporciona diferentes clases de diseños (layouts) que permiten organizar y colocar los nodos dentro de una escena.

Layouts: La clase Pane

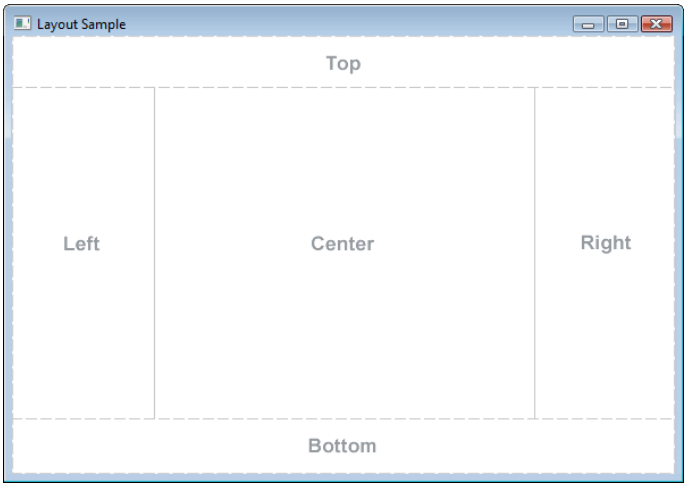
Es la clase base para todos los diseños (layouts) en JavaFX.
Proporciona un área rectangular donde los nodos se pueden colocar arbitrariamente.

Algunas subclases populares son:

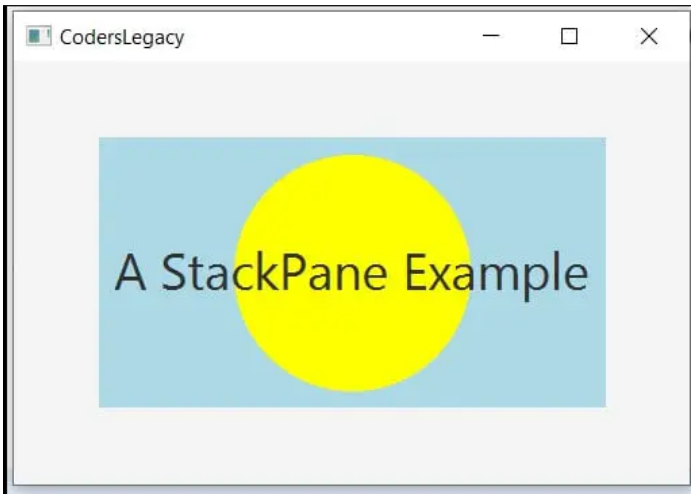
- BorderPane
- StackPane
- FlowPane
- GridPane
- HBox
- VBox
- TabPane
- ScrollPane

¿Cómo se ve cada disposición en una ventana real?

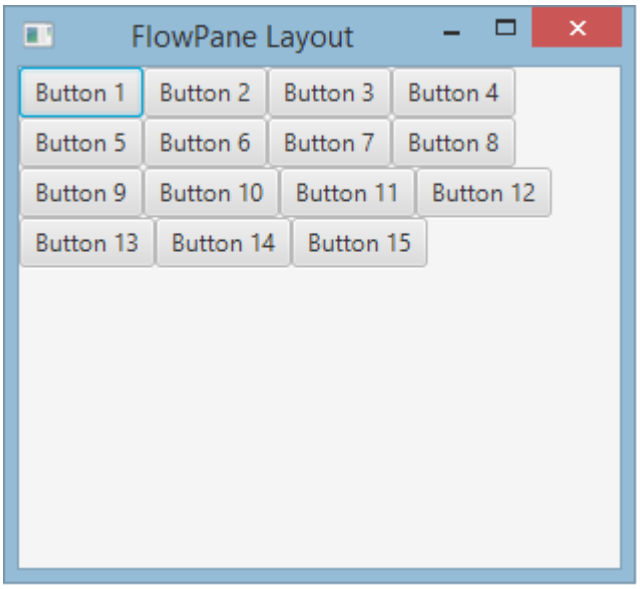
BorderPane



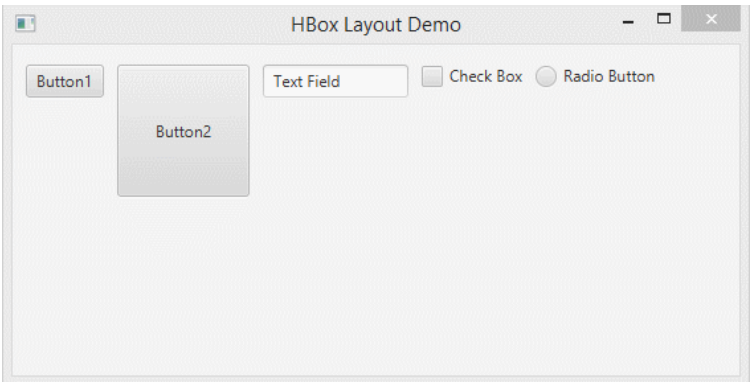
StackPane



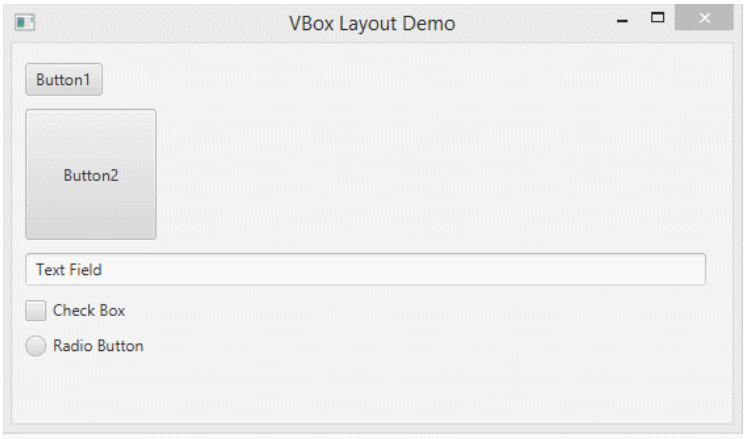
FlowPane



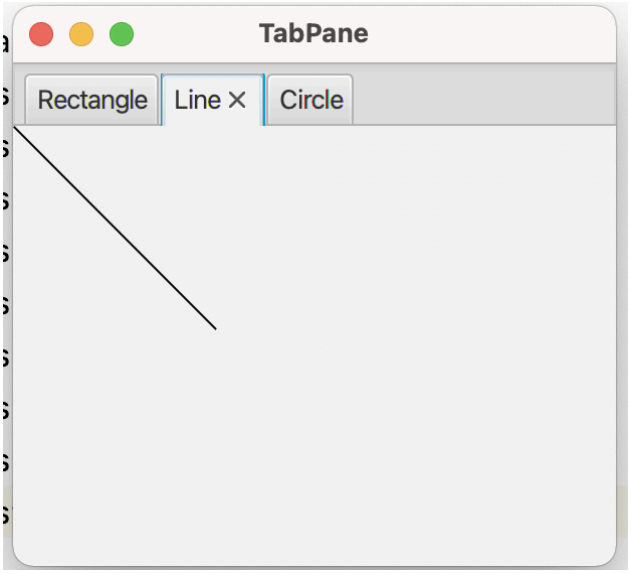
HBox



VBox



TabPane



ScrollPane