



P1. INTRODUCCIÓN A JAVAFX 8

Interfaces Persona Computador

Depto. Sistemas Informáticos y Computación

UPV

Índice

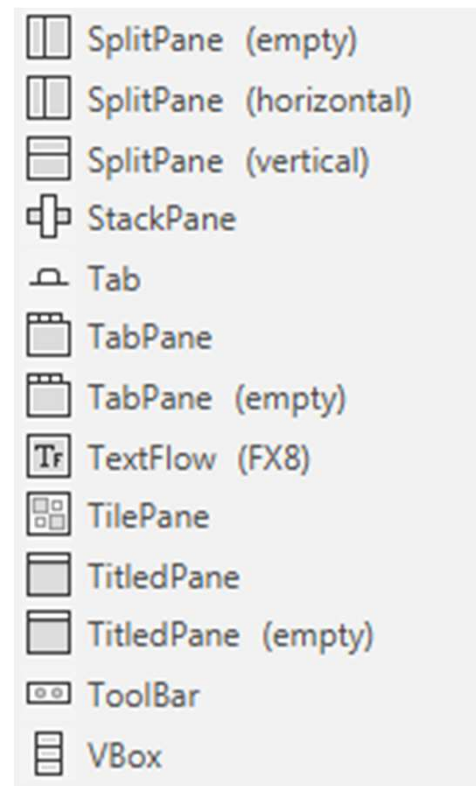
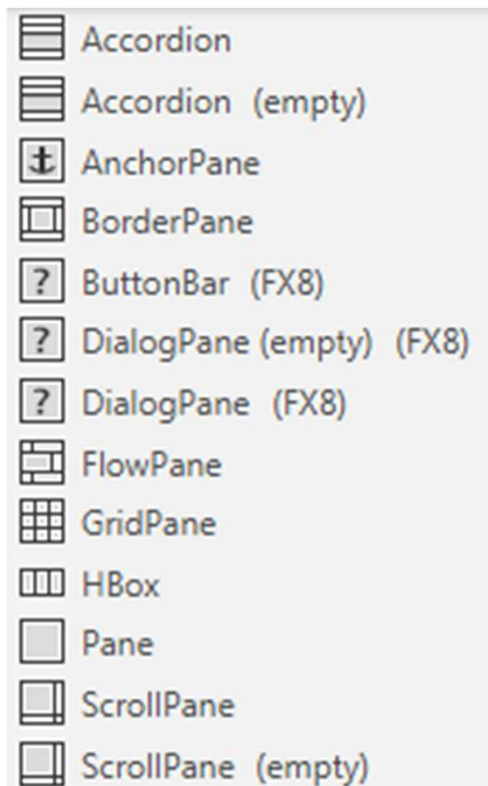
- Introducción
- Controles de JavaFX 8
- Aplicación de JavaFX por defecto en NetBeans
- Contenedores
- Inspeccionando aplicaciones JavaFX
- Ejercicio

Introducción

- En esta práctica estudiaremos los controles disponibles en JavaFX 8
- Nos centraremos en el uso de los contenedores, pues son la forma de organizar los controles de una interfaz
- Se mostrará cómo controlar el tamaño de los controles
- Se explica cómo construir los ejemplos de Ensemble en NetBeans y cómo inspeccionar aplicaciones JavaFX en ejecución para estudiar su grafo de escena

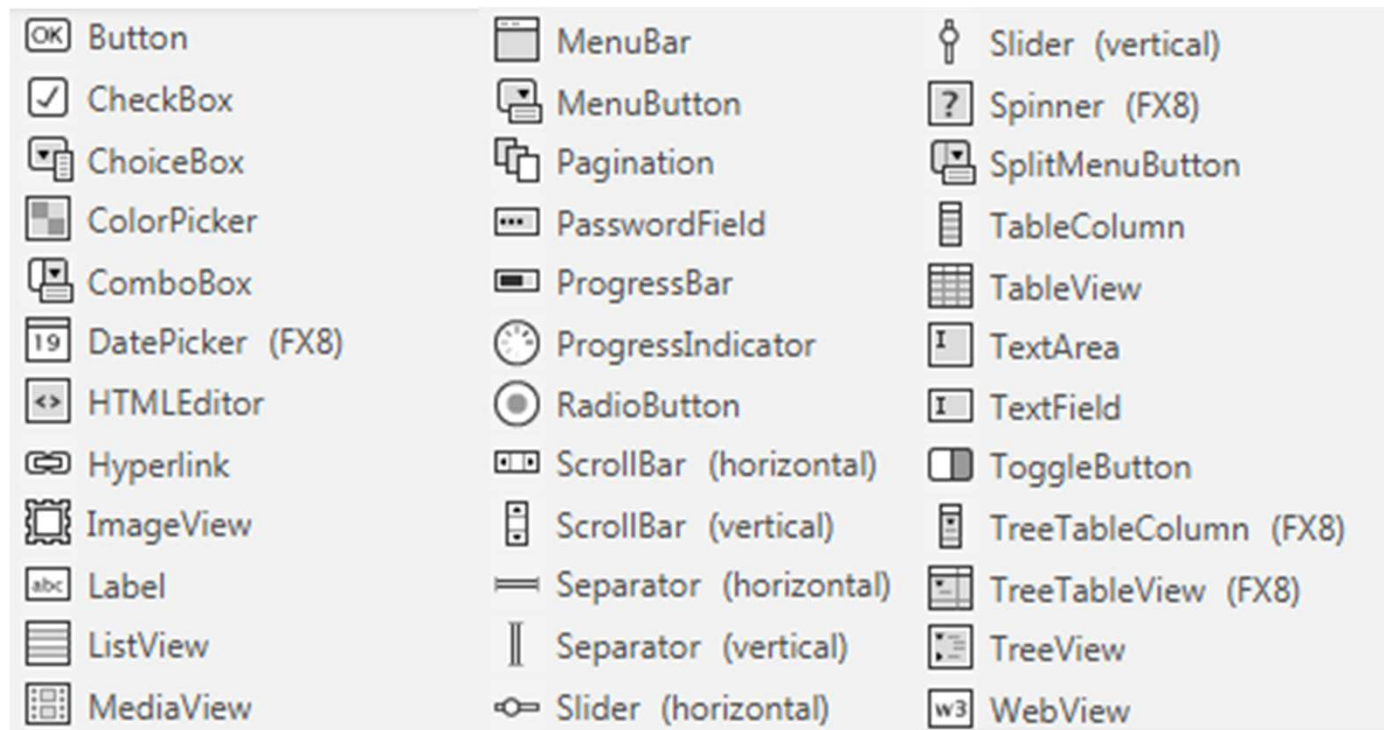
Controles de JavaFX 8

- Contenedores: contienen otros controles (y contenedores), organizados automáticamente



Controles de JavaFX 8








- Controles: implementan la interacción con el usuario











Controles de JavaFX 8

- Otros controles

Menús

-  CheckMenuItem
-  ContextMenu
-  CustomMenuItem
-  Menu
-  MenuItem
-  RadioMenuItem
-  SeparatorMenuItem







Gráficas










-  AreaChart
-  BarChart
-  BubbleChart
-  LineChart
-  PieChart
-  ScatterChart
-  StackedAreaChart
-  StackedBarChart

Gráficos 2D y 3D

-  AmbientLight (FX8)
-  ParallelCamera (FX8)
-  PerspectiveCamera (FX8)
-  PointLight (FX8)

Otros

-  Canvas
-  Group
-  Region
-  SubScene (FX8)
-  SwingNode (FX8)
-  Tooltip

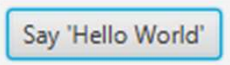
-  Arc
-  ArcTo
-  Box (FX8)
-  Circle
-  ClosePath
-  CubicCurve
-  CubicCurveTo
-  Cylinder (FX8)
-  Ellipse
-  HLineTo
-  Line
-  LineTo
-  MeshView (FX8)
-  MoveTo
-  Path
-  Polygon
-  Polyline
-  QuadCurve
-  QuadCurveTo
-  Rectangle
-  Sphere (FX8)
-  SVGPath
-  Text
-  VLineTo

Aplicación JavaFX por defecto en NetBeans

File\New Project\Java FX Application

```
import javafx.application.Application;
/* y algunos imports más */

public class HelloWorld extends Application {
    @Override
    public void start(Stage primaryStage) {
        Button btn = new Button();
        btn.setText("Say 'Hello World'");
        btn.setOnAction(new EventHandler<ActionEvent>() {
            @Override public void handle(ActionEvent event) {
                System.out.println("Hello World!");
            }
        });
        StackPane root = new StackPane();
        root.getChildren().add(btn);
        Scene scene = new Scene(root, 300, 250);
        primaryStage.setTitle("Hello World!");
        primaryStage.setScene(scene);
        primaryStage.show();
    }
    public static void main(String[] args) {
        launch(args);
    }
}
```

// Crea un botón 

// Etiqueta del botón

// Gestor de evento:

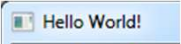
// cada vez que se pulse el botón

// se ejecutará el método handle

// Creamos un contenedor

// Añadimos el botón al contenedor

// Creamos una escena con el cont.

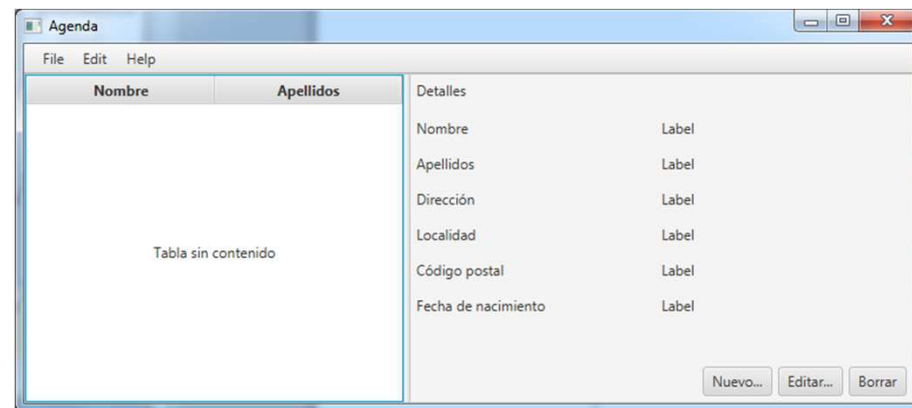
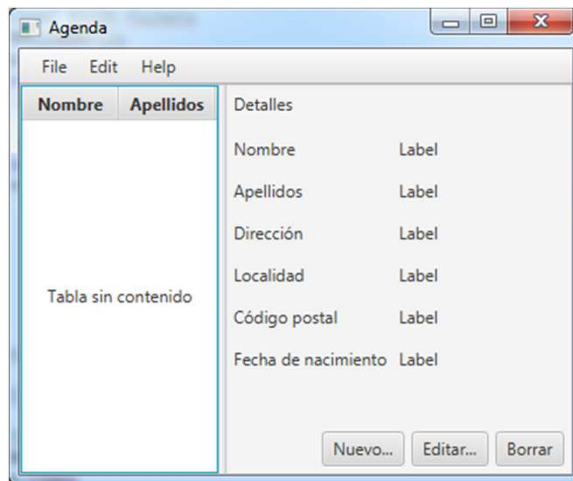
// Título de la ventana 

// Instalamos la escena en el stage

// Mostramos el escenario

Contenedores de JavaFX

- El programador puede establecer por código el tamaño y posición de los widgets en pantalla, pero es difícil de adaptarse a cambios de tamaños de la ventana
- Es más fácil (¡y obligatorio en esta asignatura!) usar contenedores para hacer que los controles se ajusten correctamente a cualquier tamaño de ventana



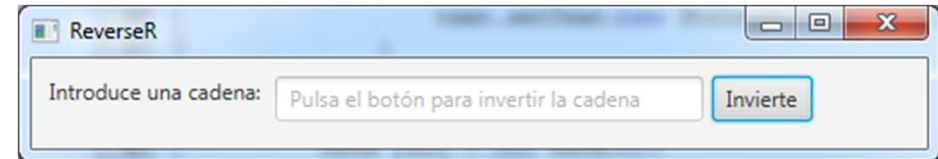
Contenedores de JavaFX

- Hay contenedores para organizar los controles por filas, columnas, apilamientos, en cuadrícula, etc.
- El contenedor recalcula automáticamente la posición y tamaño de los controles que contiene al cambiar el tamaño de la ventana

Contenedores de JavaFX

- HBox

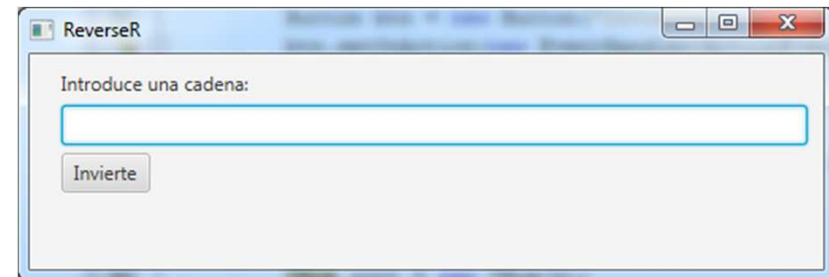
- Organiza a sus hijos en fila
- Podemos establecer un área en blanco en cada lado del HBox (*padding*), y el espacio entre los hijos (*spacing*)



```
Label lbl = new Label("Introduce una cadena:");
TextField text = new TextField(); // Campo de texto
text.setPrefColumnCount(20); // Tamaño preferido
text.setPromptText("Pulsa el botón para invertir la cadena"); // Indicación al usuario
Button btn = new Button("Invierte");
btn.setOnAction(new EventHandler<ActionEvent>() {
    @Override public void handle(ActionEvent event) {
        String tmp = text.getText();
        text.setText(new StringBuffer(tmp).reverse().toString());
    }
});
HBox root = new HBox(5); // 5 píxeles entre hijos
root.setPadding(new Insets(10)); // 10 píxeles alrededor
root.getChildren().addAll(lbl, text, btn);
```

Contenedores de JavaFX

- VBox
 - Similar a HBox, para organizar nodos en vertical
 - También define *padding* y *spacing*



// El mismo código de antes

```
VBox root = new VBox(5);
```

```
root.setPadding(new Insets(10, 10, 10, 20));
```

```
root.getChildren().addAll(lbl, text, btn);
```

// 5 píxeles entre hijos

// arriba, derecha, abajo, izq.

Contenedores de JavaFX

- **BorderPane**
- Define 5 regiones:
 - Arriba, izquierda, centro, derecha y abajo
 - Se usa para construir ventanas principales (con barra de herramientas, barra de estado, panel de navegación a la izquierda y área de trabajo en el centro)
 - Si sobra espacio, crece la zona central. Si falta, se pueden solapar
 - Puede haber regiones vacías



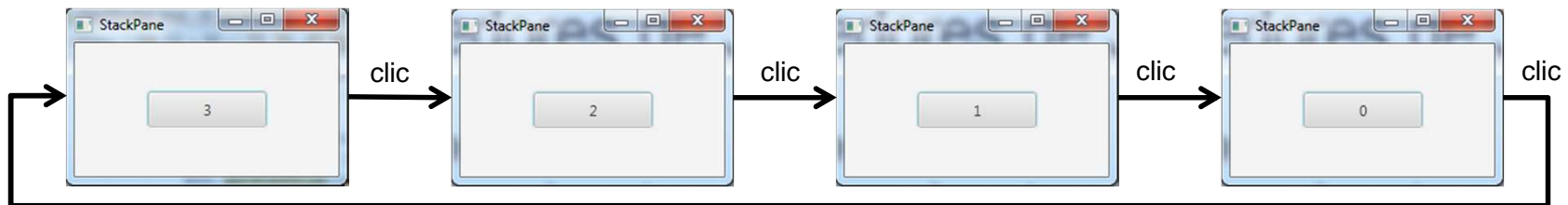
```
BorderPane root = new BorderPane();  
root.setTop(new Button("Arriba"));  
HBox group = new HBox();  
group.getChildren().addAll(  
    new Button("Abajo 1"),  
    new Button("Abajo 2"));  
root.setBottom(group);  
// y setLeft, setCenter y setRight
```

Contenedores de JavaFX

- StackPane

- Organiza sus elementos uno encima del otro
- Esto permite superponer texto sobre otros elementos (formas, imágenes, etc) o mostrar sólo un elemento de una pila

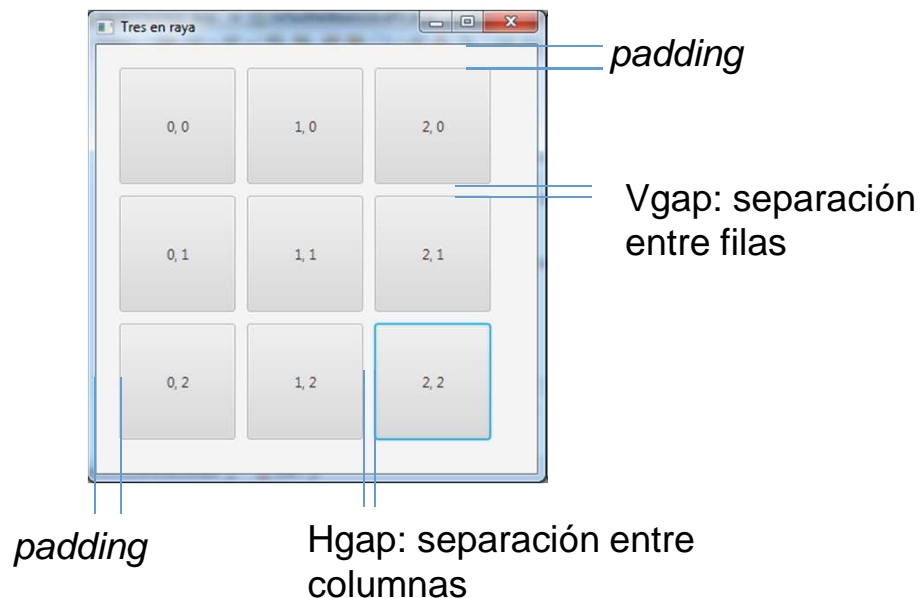
```
StackPane root = new StackPane();  
for (int i = 0; i < 4; i++) {  
    Button btn = new Button(Integer.toString(i));  
    btn.setPrefSize(100, 30);  
    btn.setOnAction(new EventHandler<ActionEvent>() {  
        @Override public void handle(ActionEvent event) {  
            ((Node)event.getSource()).toBack(); // getSource devuelve la fuente del evento  
                                                    // toBack envía al nodo al final de  
                                                    // la lista de hijos del contenedor  
        }  
    });  
    root.getChildren().add(btn);  
}
```



Contenedores de JavaFX

- GridPane

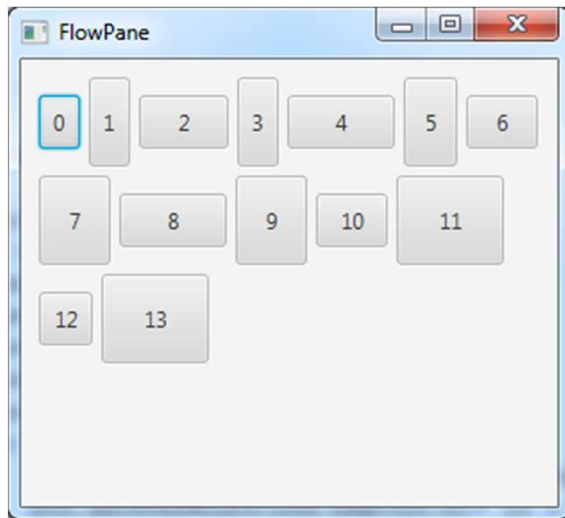
- Organiza sus elementos en una matriz de filas y columnas
- Un nodo puede expandirse a varias celdas adyacentes
- Se utiliza para crear formularios



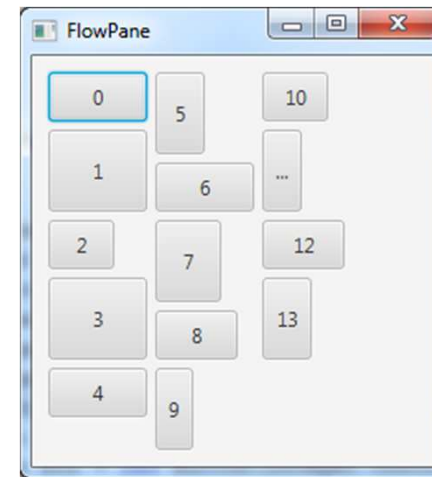
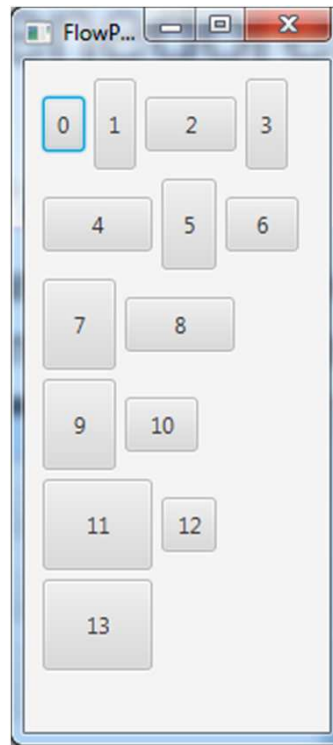
```
GridPane root = new GridPane();
for (int col = 0; col < 3; col++) {
    for (int row = 0; row < 3; row++) {
        Button btn = new Button(
            Integer.toString(col) + ", " + row);
        btn.setPrefSize(100, 100);
        root.getChildren().add(btn);
        GridPane.setConstraints(btn, col, row);
    }
}
root.setVgap(10);
root.setHgap(10);
root.setPadding(new Insets(20));
```


Contenedores de JavaFX

- FlowPane
 - Organiza los elementos secuencialmente, de izquierda a derecha o de arriba abajo, de acuerdo al tamaño del contenedor



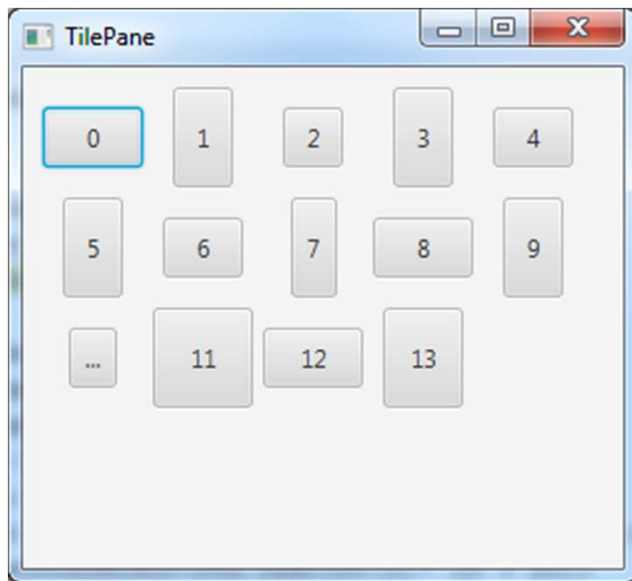
Orientation: HORIZONTAL
(por defecto)



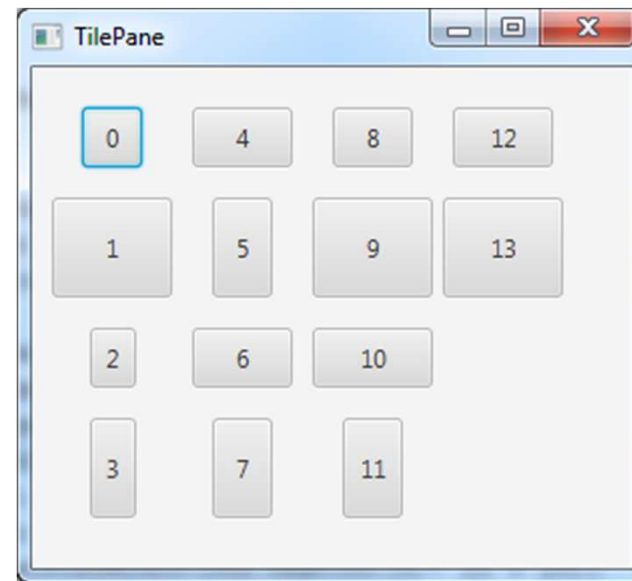
Orientation: VERTICAL

Contenedores de JavaFX

- **TilePane**
 - Es similar al FlowPane, pero organiza los elementos secuencialmente en una matriz de filas y columnas donde cada celda es del mismo tamaño



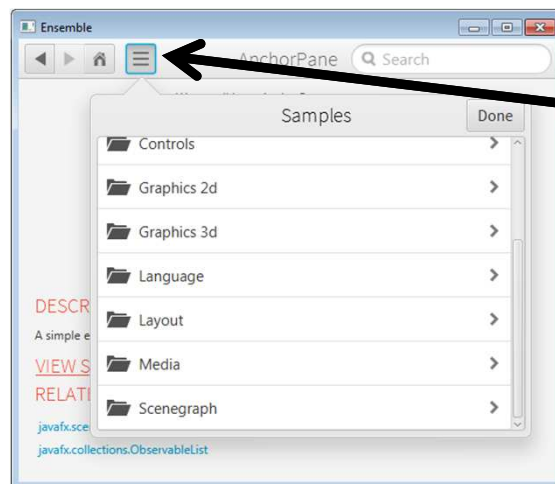
Orientation: HORIZONTAL
(por defecto)



Orientation: VERTICAL

Contenedores de JavaFX

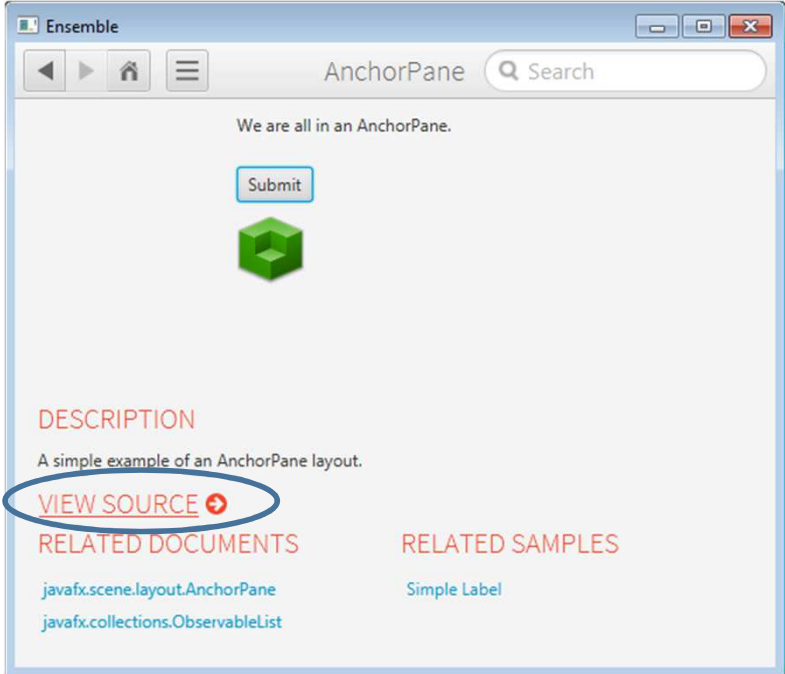
- AnchorPane
 - Permite *anclar* sus hijos a una distancia de uno de los bordes del contenedor, o al centro
 - Al cambiar el tamaño de la ventana, los nodos mantendrán su posición relativa al punto de anclaje
 - Un nodo puede estar anclado a más de un punto de anclaje
- Estudiando los ejemplos de Ensemble:



Despliega la lista por categorías y selecciona Layout y AnchorPane

Contenedores de JavaFX

- AnchorPane
 - Todas las páginas de ejemplo de Ensemble muestran:



The screenshot shows a window titled "Ensemble" with a navigation bar containing back, forward, and home icons, a search bar, and a menu icon. The main content area displays the text "We are all in an AnchorPane." above a "Submit" button and a green 3D cube. Below this, there is a "DESCRIPTION" section with the text "A simple example of an AnchorPane layout." and a "VIEW SOURCE" link circled in blue. Further down, there are sections for "RELATED DOCUMENTS" (listing `javafx.scene.layout.AnchorPane` and `javafx.collections.ObservableList`) and "RELATED SAMPLES" (listing "Simple Label").

El programa ejemplo, en ejecución (puedes interaccionar con él)

Una descripción {

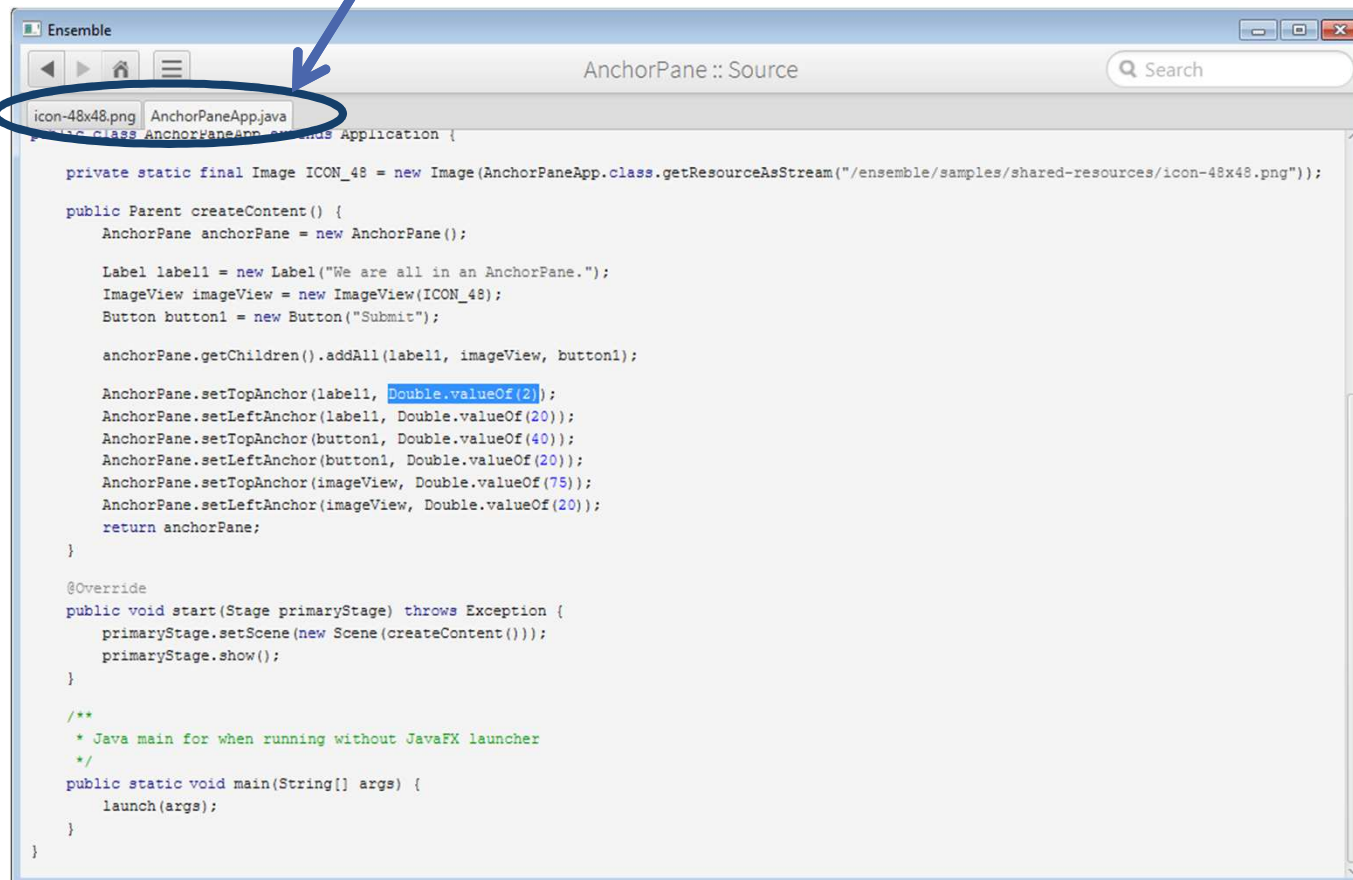
Documentación relacionada {

Ejemplos relacionados {

Contenedores de JavaFX

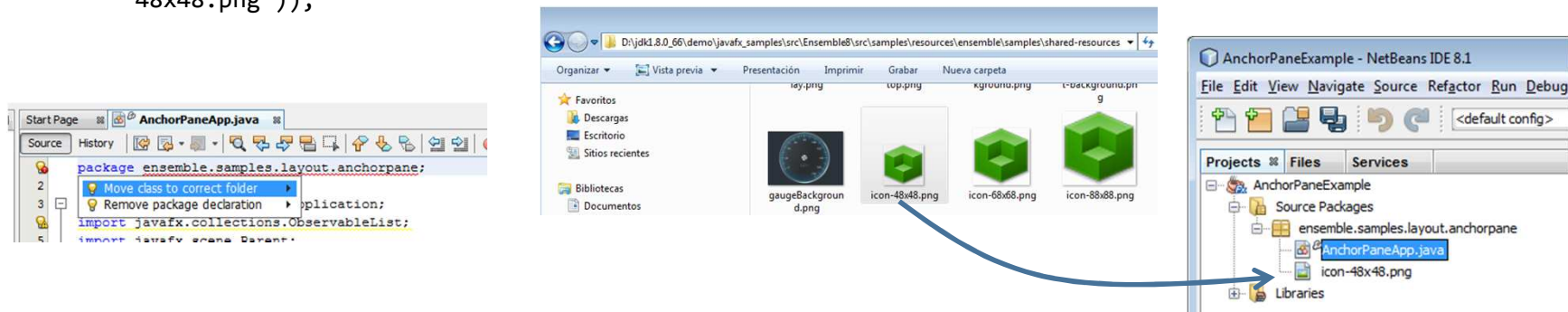
- AnchorPane

Una pestaña por cada fichero del ejemplo



Contenedores de JavaFX

- Para compilar un proyecto de Ensemble en NetBeans:
 - Crea un nuevo proyecto JavaFX Application
 - Desmarca la opción Create Application Class
 - Añade al proyecto una nueva clase Java, que se llame como la clase del ejemplo (AnchorPaneApp)
 - Copia y pega todo el código de Ensemble en el fichero recién creado
 - Tendrás que llevar la clase a su paquete
 - Si hay recursos gráficos, cópialos al paquete y ajusta su ruta en el código:
 - `private static final Image ICON_48 = new Image(AnchorPaneApp.class.getResourceAsStream("/ensemble/samples/shared-resources/icon-48x48.png"));`
 - `private static final Image ICON_48 = new Image(AnchorPaneApp.class.getResourceAsStream("icon-48x48.png"));`




Contenedores de JavaFX

Controlando el tamaño de los nodos de JavaFX

- La ventaja de utilizar los contenedores de JavaFX es que recalculan la posición/tamaño de los nodos al cambiar el tamaño de la ventana
- Los nodos cambian su tamaño de acuerdo a varios parámetros, que veremos a continuación
 - Hay nodos que no cambian su tamaño: formas, texto o grupos.
- En general, podemos especificar:
 - el tamaño de un nodo mediante su tamaño preferido (`Pref Width` y `Pref Height`)
 - la posición de un nodo mediante las propiedades de alineamiento del contenedor

Contenedores de JavaFX

Controlando el tamaño de los nodos de JavaFX

- Por defecto, los controles calculan su tamaño preferido basándose en su contenido
 - El botón se ajusta al texto que contiene
- 
- Los nodos definen tamaños mínimos y máximos
 - El máximo de un botón, por defecto está al tamaño preferido (por que normalmente no queremos que un botón crezca)
 - Otros nodos, como por ejemplo un `ListView`, sí que queremos que crezcan hasta ocupar todo el espacio disponible.
 - Podemos cambiar los tamaños:
 - Preferidos (`setPrefHeight`, `setPrefWidth` o `setPrefSize`)
 - Máximo (`setMaxHeight`, `setMaxWidth`, o `setMaxSize`)
 - Mínimo (`setMinHeight`, `setMinWidth`, o `setMinSize`)
 - Todos aceptan valores en píxeles, o `Double.MAX_VALUE`, `Control.USE_PREF_SIZE` o `Control.USE_COMPUTED_SIZE`

Contenedores de JavaFX

Controlando el tamaño de los nodos de JavaFX

- Igualando los tamaños de un grupo de nodos:

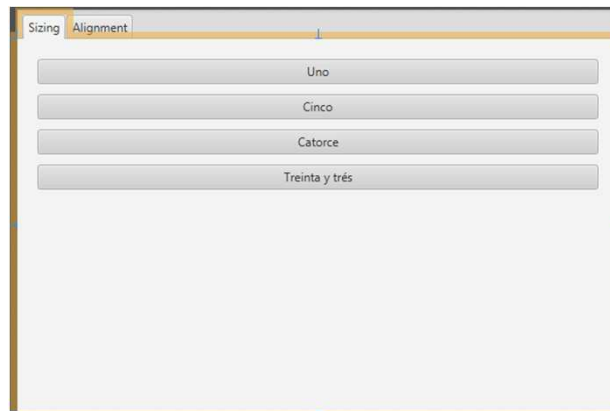
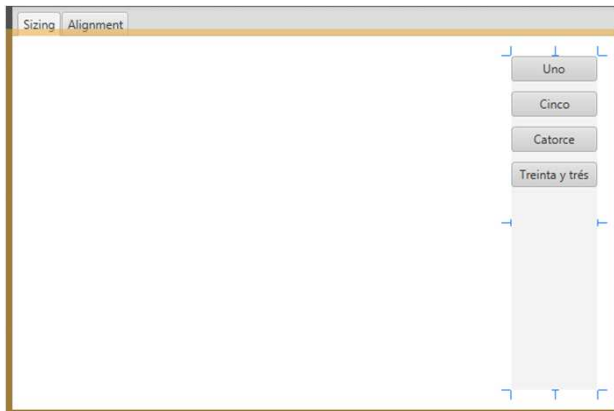
Máximo por defecto

Máximo a MAX_VALUE



```
VBox root = new VBox();  
Button b1 = new Button("Uno");  
Button b5 = new Button("Cinco");  
[...]  
b1.setMaxWidth(Double.MAX_VALUE);  
b5.setMaxWidth(Double.MAX_VALUE);  
b14.setMaxWidth(Double.MAX_VALUE);  
b33.setMaxWidth(Double.MAX_VALUE);
```

- En un BorderPane, el área central toma todo el espacio disponible; el resto sólo lo que necesitan:

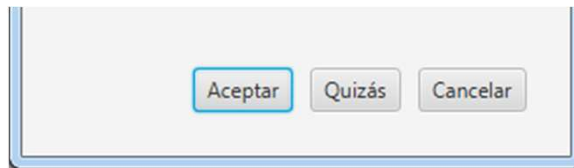


Mismo VBox que arriba, en el área derecha y en el central de un BorderPane

Contenedores de JavaFX

Controlando el tamaño de los nodos de JavaFX

- Para evitar que un nodo crezca:
 - Establecer el tamaño máximo a `Control.USE_PREF_SIZE`, o a un tamaño máximo en píxeles
- Para evitar que un nodo se haga más pequeño
 - Establecer su tamaño mínimo a `Control.USE_PREF_SIZE`

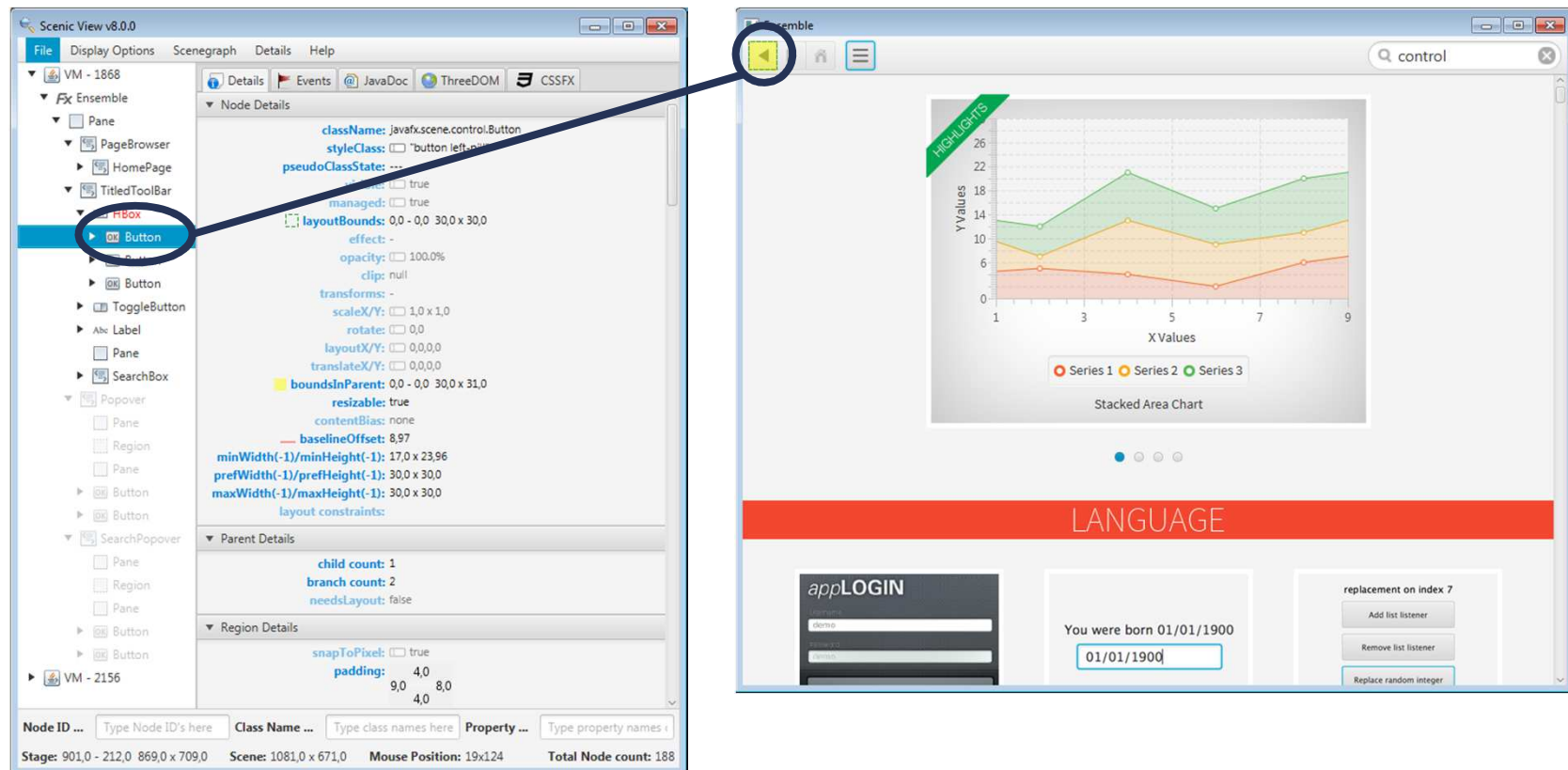


Por defecto, los botones encogen

- Para evitar que un nodo cambie de tamaño
 - Establecer los tamaños mínimo, máximo y preferido al mismo valor

Inspeccionando aplicaciones JavaFX

- Scenic View es una aplicación que permite inspeccionar las aplicaciones de JavaFX en ejecución



Ejercicio

- Construye la siguiente interfaz (no hace falta que implementes la lógica del juego)



Repasa estos vídeos antes de la siguiente sesión

- Herramientas de edición de NetBeans
 - <https://goo.gl/YszduJ>
- Exportando proyectos de NetBeans
 - <https://goo.gl/zkEURr>

Bibliografía

- JavaFX: Getting Started with JavaFX
 - <http://docs.oracle.com/javase/8/javafx/get-started-tutorial/index.html>
- JavaFX: Working with Layouts in JavaFX
 - <http://docs.oracle.com/javase/8/javafx/layout-tutorial/index.html>
- Documentación de la API de JavaFX:
 - <http://docs.oracle.com/javase/8/javafx/api/toc.htm>
- Ensemble
 - Busca “Java SE Development Kit Demos and Samples Downloads”
- Scenic View
 - <http://fxexperience.com/scenic-view/>