**BorderPane**

**Que es y para que sirve:**

Es comúnmente usado como elemento raíz de un objeto “Scene” y nos da un diseño dividido en 5 área, arriba, abajo, izquierda, derecha y centro, dentro de estas divisiones podemos colocar otros elementos de diseño o directamente algún control que necesitemos.

**Como funciona:**

JavaFX BorderPane es un contenedor que coloca a sus hijos en diferentes posiciones, como la parte inferior, superior, izquierda, derecha y central. Independientemente del valor de la propiedad visible del niño y de los hijos no administrados, este panel coloca al niño en diferentes posiciones. Java BorderPane se puede instanciar a partir de la clase javafx.scene.layout.BorderPane. Dado que cumple con todos los requisitos en el diseño de una pantalla [de interfaz gráfica de usuario](https://www.educba.com/what-is-gui/), este panel se utiliza comúnmente en el nodo raíz de la escena. Ahora, veamos la sintaxis de BorderPane.

**Sintaxis de JavaFX BorderPane**

La sintaxis de BorderPane es la que se muestra a continuación.

BorderPane bp = new BorderPane();

**Constructores de JavaFX BorderPane**

Los siguientes son los constructores de JavaFX BorderPane.

* **BorderPane()**: Se creará un diseño de BorderPane.
* **BorderPane(Nodec):**Se creará un diseño de BorderPane con el nodo c especificado como centro del panel.
* **BorderPane(Nodec, Node t, Node r, Node b, Node l):**Se creará un diseño de BorderPane con el nodo especificado c como el centro, t como la parte superior, r como la derecha, b como inferior y l como a la izquierda del panel.

### Propiedades de JavaFX BorderPane

Java BorderPane tiene varias propiedades, son:

* **inferior:**en el borde inferior del panel, se colocará el nodo.
* **superior:**en el borde superior del panel, se colocará el nodo.
* **derecha:**En el borde derecho del panel, se colocará el nodo.
* **izquierda:**en el borde izquierdo del panel, se colocará el nodo.
* **centro:**en el centro del panel, se colocará el nodo.

### Métodos de JavaFX BorderPane

Java BorderPane tiene varios métodos que realizan diferentes funcionalidades. Algunos de los métodos más utilizados se explican a continuación:

* **clearConstraints(Nodechild):**Las restricciones de BorderPane se eliminarán del nodo secundario.
* **computeMinWidth (doble altura):**El ancho mínimo de la región se calculará utilizando este método.
* **computeMinHeight(doble ancho):**La altura mínima de la región se calculará utilizando este método.
* **getAlignment(Nodo c):**Se devolverá el valor de alineación hijo.
* **setAlignment(Nodo c**, **Posvalue):**Se establecerá el valor de alineación hijo.
* **getMargin(Nodec):**se devolverá el valor de las restricciones de margen.
* **setMargin(Nodec, Pos value):**se devolverá el valor de las restricciones de margen.
* **computePrefWidth (doble altura):**se calculará el ancho preferido para la región que se necesita para la altura dada.
* **computePrefHeight(doble ancho):**se calculará la altura preferida para la región que se necesita para el ancho dado.
* **getBottom():**Se devolverá el valor de la propiedad inferior.
* **getCenter():**Se devolverá el valor de la propiedad central.
* **getLeft():**Se devolverá el valor de la propiedad izquierda.
* **getRight():**Se devolverá el valor de la propiedad correcta.
* **getTop():**Se devolverá el valor de la propiedad superior.
* **setBottom(Node v):**Se establecerá el valor v de la propiedad inferior.
* **setCenter(Node v): se**establecerá el valor v de la propiedad central.
* **setLeft(Node v):**Se establecerá el valor v de la propiedad izquierda.
* **setRight(Node v):**Se establecerá el valor de propiedad correcto v.
* **setTop(Node v):**Se establecerá el valor v de la propiedad superior.

Los bordes se pueden diseñar con fondos y bordes CSS.

BorderPane respeta los tamaños mínimos, preferidos y máximos de sus hijos. Si el rango redimensionable del niño evita que se redimensione para que se ajuste a su posición, se alineará en relación con el espacio utilizando una alineación predeterminada de la siguiente manera:

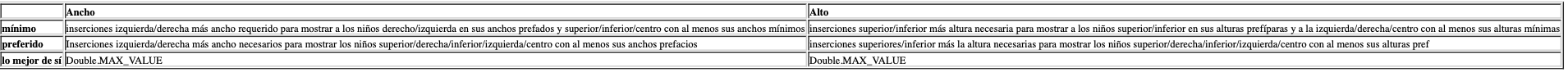
* Arriba: Pos.TOP\_LEFT
* Parte inferior: Post.BOTTOM\_LEFT
* Izquierda: Pos.TOP\_LEFT
* Derecha: Pos.TOP.RIGHT
* Centro: Pos.CENTER

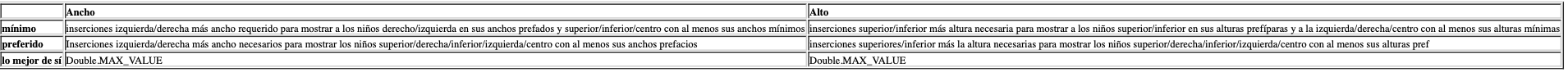
Consultar “Restricciones de diseño opcionales” sobre cómo personalizar estas alineaciones.

BorderPane establece cada niño establecido en las cinco posiciones, independientemente del valor de la propiedad visible del niño; los hijos no administrados son ignorados.

Rango redimensionable:

BorderPane se utiliza comúnmente como raíz de una Scene, en cuyo caso su tamaño hará un seguimiento del tamaño de la escena. Si la aplicación no ha establecido directamente el tamaño de la escena o el escenario, el tamaño de la escena se inicializará con el tamaño preferido del panel de borde. Sin embargo, si un panel de borde tiene un padre que no sea la escena, ese padre cambiará el tamaño del panel de borde dentro del rango de tamaño del panel de borde durante el diseño. De forma predeterminada, el panel de borde calcula este rango en función de su contenido como se describe en la siguiente tabla.





El ancho y la altura máximos ilimitados de un panel de borde son una indicación para el padre de que se puede cambiar de tamaño más allá de su tamaño preferido para llenar cualquier espacio que se le asigne.

BordePane proporciona propiedades para establecer el rango de tamaño directamente. Estas propiedades por defecto son el valor centinela Region.USE\_COMPUTED\_SIZE, sin embargo, la aplicación puede establecerlas en otros valores según sea necesario:

bordePane.setPrefSize(500, 400);

Las aplicaciones pueden restaurar los valores calculados estableciendo estas propiedades de nuevo en Region.USE\_COMPUTED\_SIZE.

BorderPane no recorta de su contenido de forma predeterminada, por lo que es posible que los límites de los niños se extiendan fuera de sus propios límites si el tamaño de un niño evita que se ajuste dentro de su espacio.

Restricciones de diseño opcionales:

Una aplicación puede establecer restricciones en niños individuales para personalizar el diseño de BorderPane. Para cada restricción, BorderPane proporciona un método estático para establecerla en el hijo.



Ejemplo:

ListView list = new ListView();

BorderPane.setAlignment(list, Pos.TOP\_LEFT);

BorderPane.setMargin(list, new Insets(12,12,12,12));

borderPane.setCenter(list);

**Como se usa y ejemplo Pràctico:**

package com.example.bordepanejavafx5;  
  
import javafx.application.Application;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.Label;  
import javafx.scene.layout.BorderPane;  
import javafx.scene.text.Text;  
import javafx.stage.Stage;  
  
public class Label\_Test extends Application {  
  
 @Override  
 public void start(Stage stage)  
 {  
 // Create the Text Nodes  
 Text centerText = new Text("Center");  
 Text topText = new Text("Top");  
 Text rightText = new Text("Right");  
 Text bottomText = new Text("Bottom");  
 Text leftText = new Text("Left");  
  
 // Set the alignment of the Top Text to Center  
 BorderPane.*setAlignment*(topText, Pos.*TOP\_CENTER*);  
 // Set the alignment of the Bottom Text to Center  
 BorderPane.*setAlignment*(bottomText,Pos.*BOTTOM\_CENTER*);  
 // Set the alignment of the Left Text to Center  
 BorderPane.*setAlignment*(leftText,Pos.*CENTER\_LEFT*);  
 // Set the alignment of the Right Text to Center  
 BorderPane.*setAlignment*(rightText,Pos.*CENTER\_RIGHT*);  
  
 // Create a BorderPane with a Text node in each of the five regions  
 BorderPane root = new BorderPane(centerText, topText, rightText, bottomText, leftText);  
 // Set the Size of the VBox  
 root.setPrefSize(400, 400);  
 // Set the Style-properties of the BorderPane  
 root.setStyle("-fx-padding: 10;" +  
 "-fx-border-style: solid inside;" +  
 "-fx-border-width: 2;" +  
 "-fx-border-insets: 5;" +  
 "-fx-border-radius: 5;" +  
 "-fx-border-color: blue;");  
  
 // Create the Scene  
 Scene scene = new Scene(root);  
 // Add the scene to the Stage  
 stage.setScene(scene);  
 // Set the title of the Stage  
 stage.setTitle("A simple BorderPane Example");  
 // Display the Stage  
 stage.show();  
 }  
  
 public static void main(String[] args) {  
 launch(args);  
 }  
}