Tutoriel Interface graphique avec JavaFX

Hello world Application

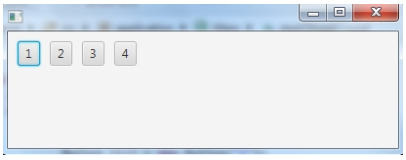
```
import javafx.application.Application;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.Label;
import javafx.scene.image.Image;
import javafx.stage.Stage;
public class MyApplication extends Application {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
	    Application.launch(args);
	}

	//@Override
	public void start(Stage primaryStage) throws Exception {
	    primaryStage.setTitle("Hello world Application");
	    primaryStage.setWidth(300);
	    primaryStage.setHeight(200);
	    Label helloWorldLabel = new Label("Hello world!");
	    helloWorldLabel.setAlignment(Pos.CENTER);
	    Scene primaryScene = new Scene(helloWorldLabel);
	    primaryStage.setScene(primaryScene);
	    primaryStage.show();
	}
}
```

Layouts

Un Layout (container ou gestionnaire de mise en page) est un container graphique qui hérite de la classe javafx.scene.layout.Pane. Il permet d'afficher un groupe de widgets (ou d'autres containers) suivant une disposition qui lui est propre.

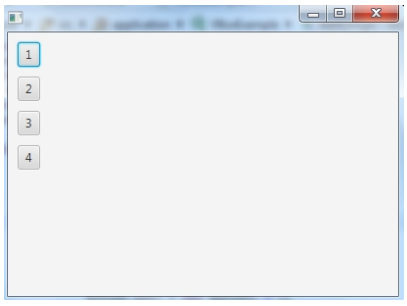
HBox : est un conteneur (container), qui arrange les sous-composants sur une seule ligne.

```
public class Main extends Application {
@Override
public void start(Stage primaryStage) {
try {
HBox hbox = new HBox();
Button btn1 = new Button("1");
Button btn2 = new Button("2");
Button btn3 = new Button("3");
Button btn4 = new Button("4");
hbox.setSpacing(10);
hbox.setPadding(new Insets(10, 10, 10, 10));
hbox.getChildren().addAll(btn1, btn2, btn3, btn4);


BorderPane root = new BorderPane();
root.setTop(hbox);
Scene scene = new Scene(root,400,400);
scene.getStylesheets().add(getClass().getResource("application.css").toExte
rnalForm());
primaryStage.setScene(scene);
primaryStage.show();

} catch(Exception e) {
e.printStackTrace();
}
}

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

}
```



VBox : est un conteneur (container), qui arrange les sous-composants sur une seule colonne.

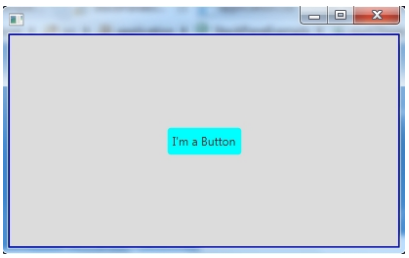
```
public class VBoxExample extends Application {


public void start(Stage primaryStage) {
try {
VBox vbox = new VBox();
vbox.setPadding(new Insets(10, 10, 10, 10));
vbox.setSpacing(10);
Button btn1 = new Button("1");
Button btn2 = new Button("2");
Button btn3 = new Button("3");
Button btn4 = new Button("4");
vbox.getChildren().addAll(btn1, btn2, btn3, btn4);

BorderPane root = new BorderPane();
```



```
root.setTop(vbox);
Scene scene = new Scene(root,400,400);
scene.getStylesheets().add(getClass().getResource("application.css").toExte
rnalForm());
primaryStage.setScene(scene);
primaryStage.show();

} catch(Exception e) {
e.printStackTrace();
}
}

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

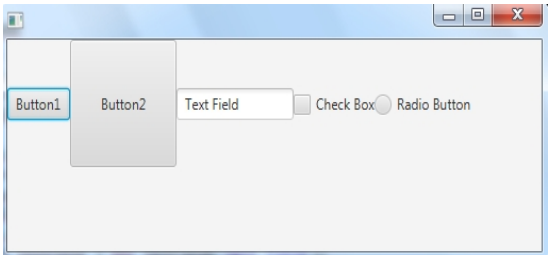
StackPane est un conteneur qui peut contenir différents composants d'interface, des sous-composants empilés à d'autres et à un moment donné, vous ne pouvez voir que le sous-composant situé sur le dessus de Stack.

```
StackPane     stackPane = new StackPane();
// Add Label to StackPane
Label label = new Label("I'm a Label");
label.setStyle("-fx-background-color:yellow");
label.setPadding(new Insets(5,5,5,5));
stackPane.getChildren().add(label);
// Add Button to StackPane
Button button = new Button("I'm a Button");
button.setStyle("-fx-background-color: cyan");
button.setPadding(new Insets(5,5,5,5));
stackPane.getChildren().add(button);
stackPane.setPrefSize(300, 150);
stackPane.setStyle("-fx-background-color: Gainsboro;-fx-border-
color:blue;");
Scene scene = new Scene(stackPane,400,400);
scene.getStylesheets().add(getClass().getResource("application.css").toExte
rnalForm());
primaryStage.setScene(scene);
primaryStage.show();
```



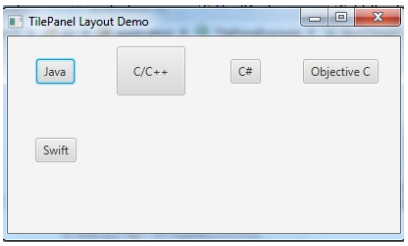
FlowPane est un conteneur (Container), il peut contenir des **Control** ou des autres conteneurs. Il organise les sous-composants consécutifs sur une ligne et appuie automatiquement les sous-composants jusqu'à la ligne suivante si la ligne actuelle est remplie.

```
FlowPane root = new FlowPane();
// Button 1
Button button1= new Button("Button1");
root.getChildren().add(button1);

// Button 2
Button button2 = new Button("Button2");
button2.setPrefSize(100, 100);
root.getChildren().add(button2);
// TextField
TextField textField = new TextField("Text Field");
textField.setPrefWidth(110);

root.getChildren().add(textField);

// CheckBox
CheckBox checkBox = new CheckBox("Check Box");

root.getChildren().add(checkBox);

// RadioButton
RadioButton radioButton = new RadioButton("Radio Button");
root.getChildren().add(radioButton);
Scene scene = new Scene(root,400,400);
scene.getStylesheets().add(getClass().getResource("application.css").toExte
rnalForm());
primaryStage.setScene(scene);
primaryStage.show();
```



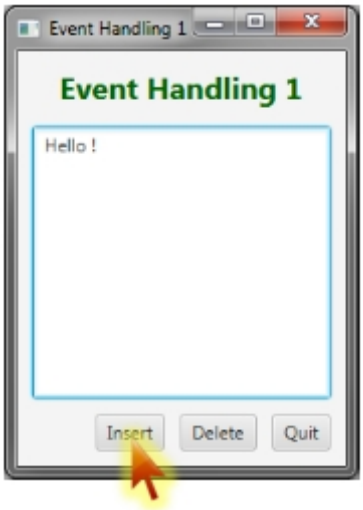
TilePane est un conteneur (Container), **TilePane** est similaire à **FlowPane**. Il arrange les sous-composants consécutifs sur une ligne, et pousse automatiquement les sous-composants vers la ligne suivante si la ligne actuelle est remplie. Cependant, cela diffère de **FlowPane** parce que les sous-composants se trouvent sur la même taille de cellule.

```
TilePane root = new TilePane();

root.setPadding(new Insets(10,10,10,10));
root.setHgap(20);
root.setVgap(30);

Button button = new Button("Java");
root.getChildren().add(button);

// Short Button
Button button1 = new Button("C/C++");
button1.setPrefSize(70, 50);
root.getChildren().add(button1);

// Short Button
Button button2 = new Button("C#");

root.getChildren().add(button2);
```



```
// Button
Button longButton3 = new Button("Objective C");
root.getChildren().add(longButton3);

// Button
Button button4 = new Button("Swift");
root.getChildren().add(button4);

Scene scene = new Scene(root,400,400);
scene.getStylesheets().add(getClass().getResource("application.css").toExte
rnalForm());
primaryStage.setTitle("TilePanel Layout Demo ");

primaryStage.setScene(scene);
primaryStage.show();
```

Gestion des événements




```
public class EventHandligExample extends Application {

public void start(Stage primaryStage) {
try {


BorderPane root = new BorderPane();
HBox btnPanel = new HBox(10);
Label      lblTitle = new Label("Event Handling");
TextArea txaMsg = new TextArea();
Button btnInsert = new Button("Insert");
Button btnDelete = new Button("Delete");
Button     btnQuit = new Button("Quit");
primaryStage.setTitle("Event Handling");
root.setPadding(new Insets(10));
//--- Title
lblTitle.setFont(Font.font("System", FontWeight.BOLD, 20));
lblTitle.setTextFill(Color.DARKGREEN);
BorderPane.setAlignment(lblTitle, Pos.CENTER);
BorderPane.setMargin(lblTitle, new Insets(0, 0, 10, 0));
root.setTop(lblTitle);
//--- Text-Area
txaMsg.setWrapText(true);
txaMsg.setPrefColumnCount(15);
txaMsg.setPrefRowCount(10);
root.setCenter(txaMsg);
//--- Button Panel
btnPanel.getChildren().add(btnInsert);
btnPanel.getChildren().add(btnDelete);
btnPanel.getChildren().add(btnQuit);
btnPanel.setAlignment(Pos.CENTER_RIGHT);
btnPanel.setPadding(new Insets(10, 0, 0, 0));
root.setBottom(btnPanel);
//--- Button Insert
InsertButtonController insertCtrl = new InsertButtonController(txaMsg);
btnInsert.addEventHandler(ActionEvent.ACTION, insertCtrl);
```

```
//--- Button Delete
btnDelete.addEventHandler(ActionEvent.ACTION,new
EventHandler<ActionEvent>() {
@Override
public void handle(ActionEvent event) {
txaMsg.deletePreviousChar();
}
});
//--- Button Delete
btnQuit.setOnAction(event -> {
Platform.exit();
});

Scene scene = new Scene(root);
primaryStage.setScene(scene);
primaryStage.show();
} catch(Exception e) {
e.printStackTrace();
}
}

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

Classe InsertButtonController

```
public class InsertButtonController implements EventHandler<ActionEvent> {
private TextArea tArea;
//--- Constructeur ---------------------------------
public InsertButtonController(TextArea tArea) {
this.tArea = tArea;
}
//--- Code exécuté lorsque l'événement survient ----
public void handle(ActionEvent event) {
tArea.appendText("A");
}
}
```