# 1 GUI

*JavaFX* – first it was external library but now is recommended technology to create GUI in new desktop applications.

## 1.1 Windows

*Application* class is main class for JavaFX application and it is responsible for displaying windows.

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

public class FXTests extends Application {

    @Override
    public void start(Stage primaryStage) {

        StackPane root = new StackPane();
        Scene scene = new Scene(root, 300, 250);
        primaryStage.setTitle("Hello World!");
        primaryStage.setScene(scene);
        primaryStage.show();
    }

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

}
```

## 1.2 Components

There are numerous components that can be included in windows:

- Label
- Button
- ComboBox
- ProgressBar
- ...

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

public class FXTests extends Application {

    @Override
    public void start(Stage primaryStage) {
        StackPane root = new StackPane();

        Button btn = new Button();
        btn.setText("Hello");
        root.getChildren().add(btn);

        /*Label lbl = new Label("Hello");
        root.getChildren().add(lbl);*/

        Scene scene = new Scene(root, 300, 250);

        primaryStage.setTitle("Hello World!");
        primaryStage.setScene(scene);
        primaryStage.show();
    }

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

}
```

## 1.3 Layout managers

Layout managers are responsible for proper distribution of components in window:

- StackPane
- BorderPane
- HBox
- VBox
- GridPane
- ...

### 1.3.1 BorderPane

```
@Override
public void start(Stage primaryStage) {
  BorderPane root = new BorderPane();

  Button btn = new Button();
  Button btn2 = new Button("Button");
  btn.setText("Say 'Hello World'");
  Label lbl = new Label("Hello");

  root.setTop(btn);
  root.setRight(btn2);
  root.setLeft(lbl);

  Scene scene = new Scene(root, 300, 250);
  primaryStage.setTitle("Hello World!");
  primaryStage.setScene(scene);
  primaryStage.show();
}
```

### 1.3.2 HBox

```
@Override
public void start(Stage primaryStage) {
  HBox root = new HBox();

  Button btn = new Button();
  Button btn2 = new Button("Button");
  btn.setText("Say 'Hello World'");
  Label lbl = new Label("Hello");

  root.getChildren().add(btn);
  root.getChildren().add(btn2);
  root.getChildren().add(lbl);

  Scene scene = new Scene(root, 300, 250);
  primaryStage.setTitle("Hello World!");
  primaryStage.setScene(scene);
  primaryStage.show();
}
```

### 1.3.3 VBox

```
@Override
public void start(Stage primaryStage) {
  VBox root = new VBox();

  Button btn = new Button();
  Button btn2 = new Button("Button");
  btn.setText("Say 'Hello World'");
  Label lbl = new Label("Hello");

  root.getChildren().add(btn);
  root.getChildren().add(btn2);
  root.getChildren().add(lbl);

  Scene scene = new Scene(root, 300, 250);
  primaryStage.setTitle("Hello World!");
```

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

# 2 Handling events

```
Button btn = new Button();
btn.setOnAction(new EventHandler<ActionEvent>() {

  @Override
  public void handle(ActionEvent event) {
    System.out.println("Cliked! :)");
  }
});
```

# 3 FXML

To declare user interface FXML language can be used. In that matter the application logic can be split from the presentation.

## 3.1 FXML file

```
<?xml version="1.0" encoding="UTF-8"?>
<?import java.lang.*?>
<?import java.util.*?>
<?import javafx.scene.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<VBox id="AnchorPane" prefHeight="400.0" prefWidth="600.0" xmlns:fx="http://javafx.com/↩
    fxml/1" fx:controller="com.adeik.fxtests.FXMLController">

<Label text="Label" id="label" minHeight="16" minWidth="69" prefHeight="16" prefWidth="↩
    69" fx:id="label" />
<Button text="Button" onAction="#buttonHandler" />
<TextField text="TextField" fx:id="textField" />

</VBox>
```

## 3.2 Controller

Controller object is responsible for handling user interface.

```
package com.adeik.fxtests;

import java.net.URL;
import java.util.ResourceBundle;
import javafx.event.Event;
import javafx.fxml.FXML;
import javafx.fxml.Initializable;
```

```
import javafx.scene.control.TextField;

public class FXMLController implements Initializable {

    @FXML
    private TextField textField;

    @Override
    public void initialize(URL url, ResourceBundle rb) {
        // TODO
    }

    @FXML
    protected void buttonHandler(Event e){
        textField.setText("Hello!");
    }

}
```

## 3.3 Window

```
package com.adeik.fxtests;

import java.io.IOException;
import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.stage.Stage;

public class FXMLTest extends Application {

    @Override
    public void start(Stage primaryStage) {
        try {
            Pane myPane = (Pane)FXMLLoader.load(FXMLTest.class.getResource("/com/adeik/fxtests↩
     /FXML.fxml"));
            Scene myScene = new Scene(myPane);
            primaryStage.setScene(myScene);
            primaryStage.show();
        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }

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

}
```

# 4 Tasks to complete

1. Make familiar with SceneBuilder programme

2. Make familiar with documentation of LayoutManagers (JavaFX Layout Managers)
3. Create the window with components that represents user registration form.
4. Handle events with EventHandlers
5. Use lambdas to handle events
6. Create layout in FXML file