Graphical User Interfaces

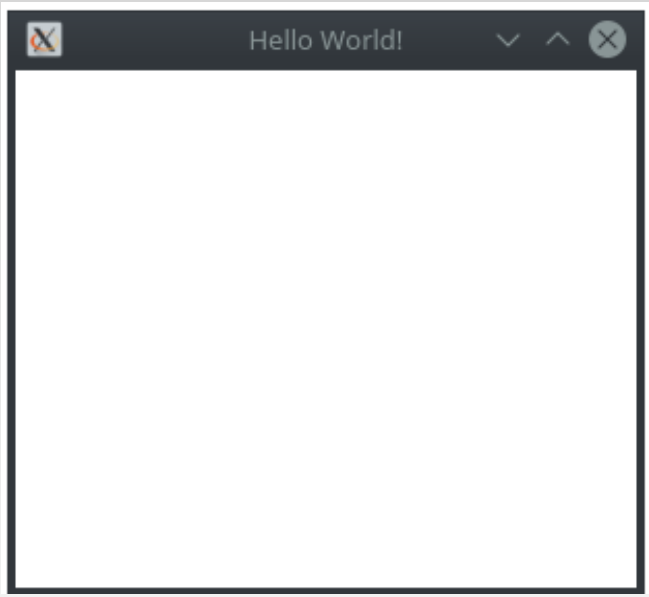
Adam Krechowicz

Graphical libraries

- AWT – Abstract Window Toolkit
- Swing
- SWT – Standard Widget Toolkit
- JavaFX

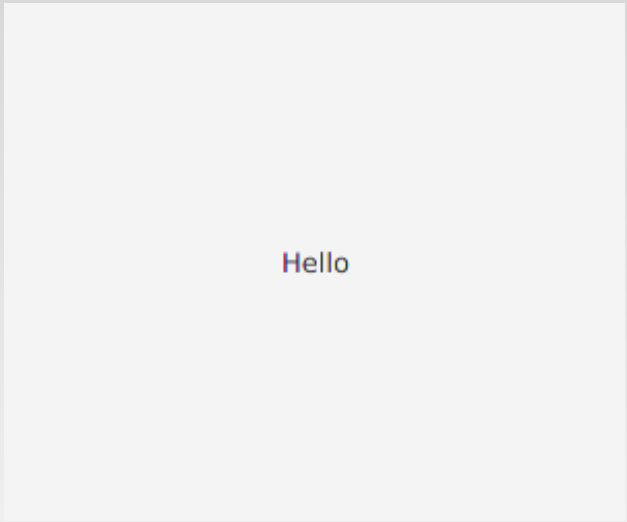
Window

```
1   public class FXTests extends Application {
2
3   @Override
4   public void start(Stage primaryStage) {
5
6       StackPane root = new StackPane();
7       Scene scene = new Scene(root, 300, ↩
        250);
8       primaryStage.setTitle("Hello World!");
9       primaryStage.setScene(scene);
10      primaryStage.show();
11  }
12
13  public static void main(String[] args) {
14      launch(args);
15  }
16
17 }
18
```



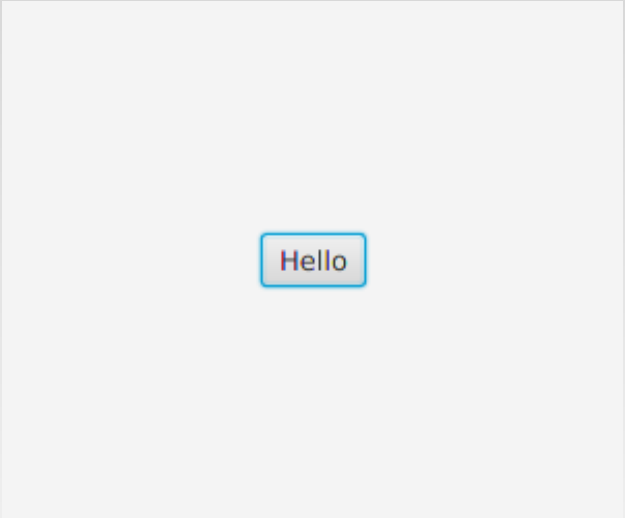
Basic components

```
1  Label lbl = new Label("Hello");
2  root.getChildren().add(lbl);
3
```



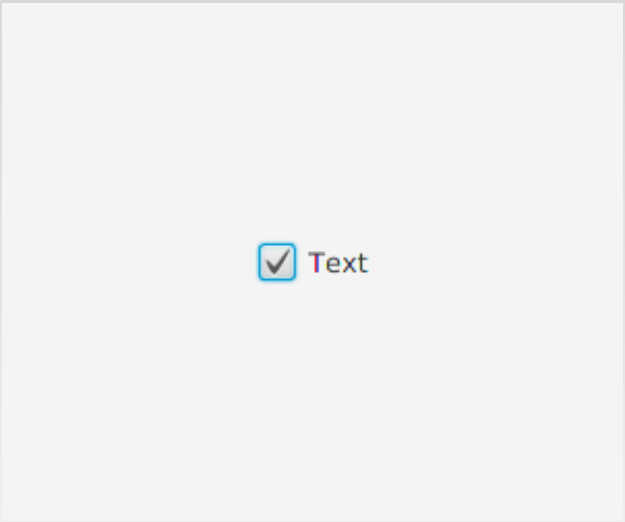
Basic components

```
1  Button btn = new Button();
2  btn.setText("Hello");
3  root.getChildren().add(btn);
4
```



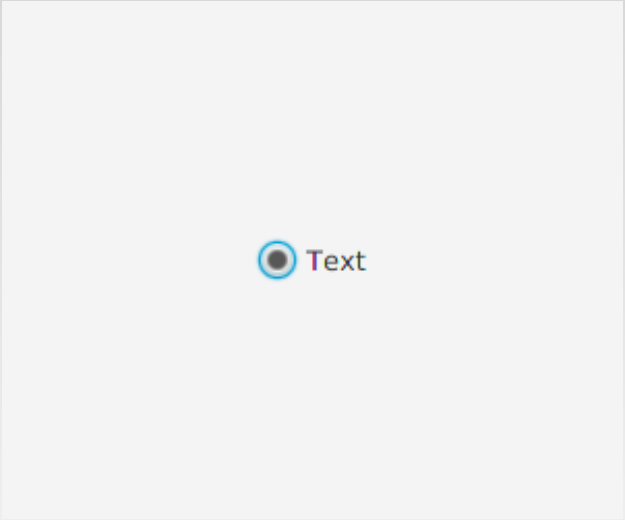
Basic components

```
1  root.getChildren().add(new CheckBox("Text"));
2
```



Basic components

```
1  root.getChildren().add(new RadioButton("Text")↩
       );
2
```



Basic components

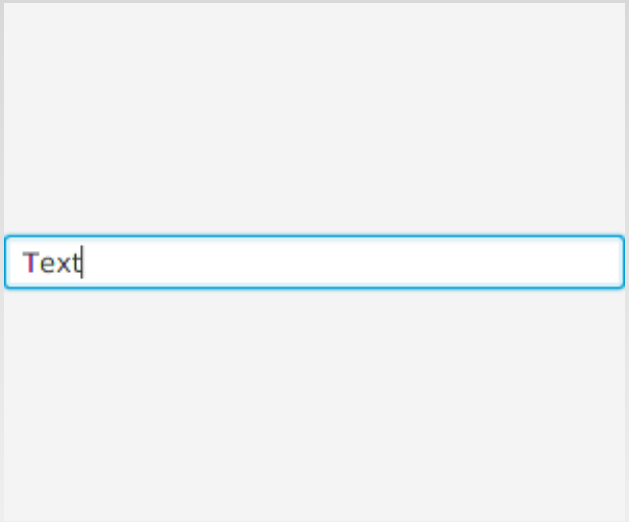
```
1  root.getChildren().add(new new ToggleButton("↩
       Text"));
2
```

Basic components

```
1  ChoiceBox cb = new ChoiceBox(FXCollections.↩
       observableArrayList("jeden", "dwa", "trzy"↩
       ));
2  root.getChildren().add(cb);
3
```

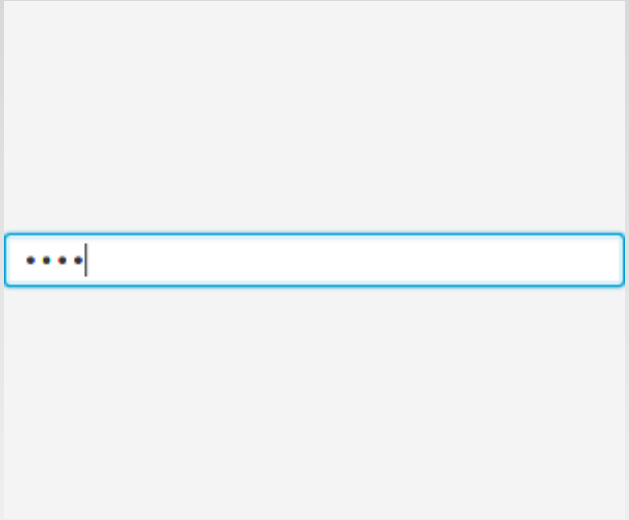
Basic components

```
1  root.getChildren().add(new TextField("Text"));
2
```



Basic components

```
1  root.getChildren().add(new PasswordField());
2
```

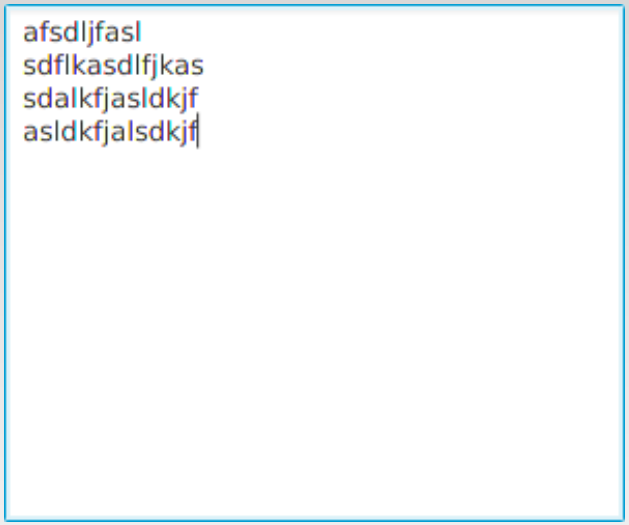


Basic components

```
1  root.getChildren().add(new ScrollBar());
2
```

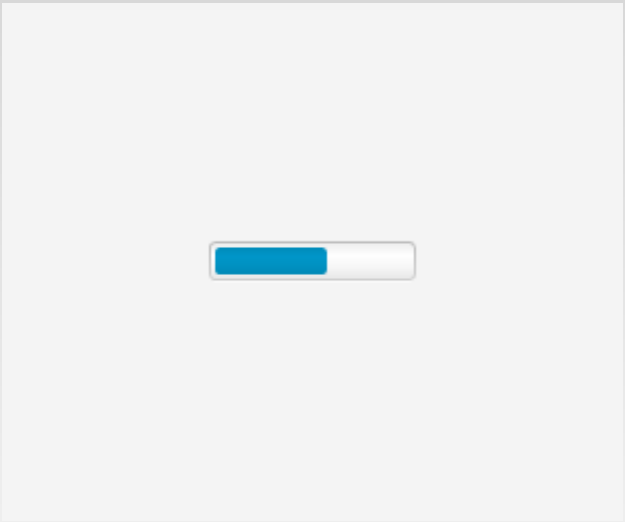
Basic components

```
1  root.getChildren().add(new TextArea());
2
```



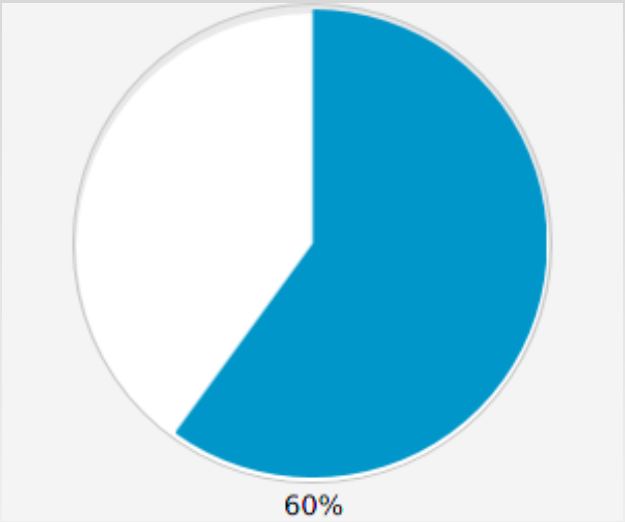
Basic components

```
1  root.getChildren().add(new ProgressBar(0.6));
2
```



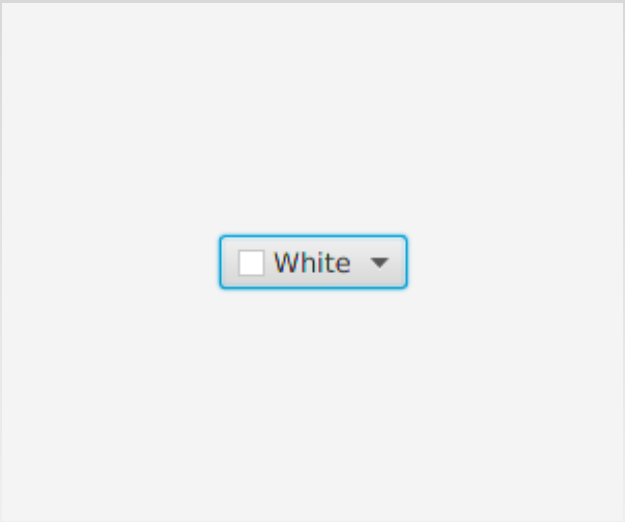
Basic components

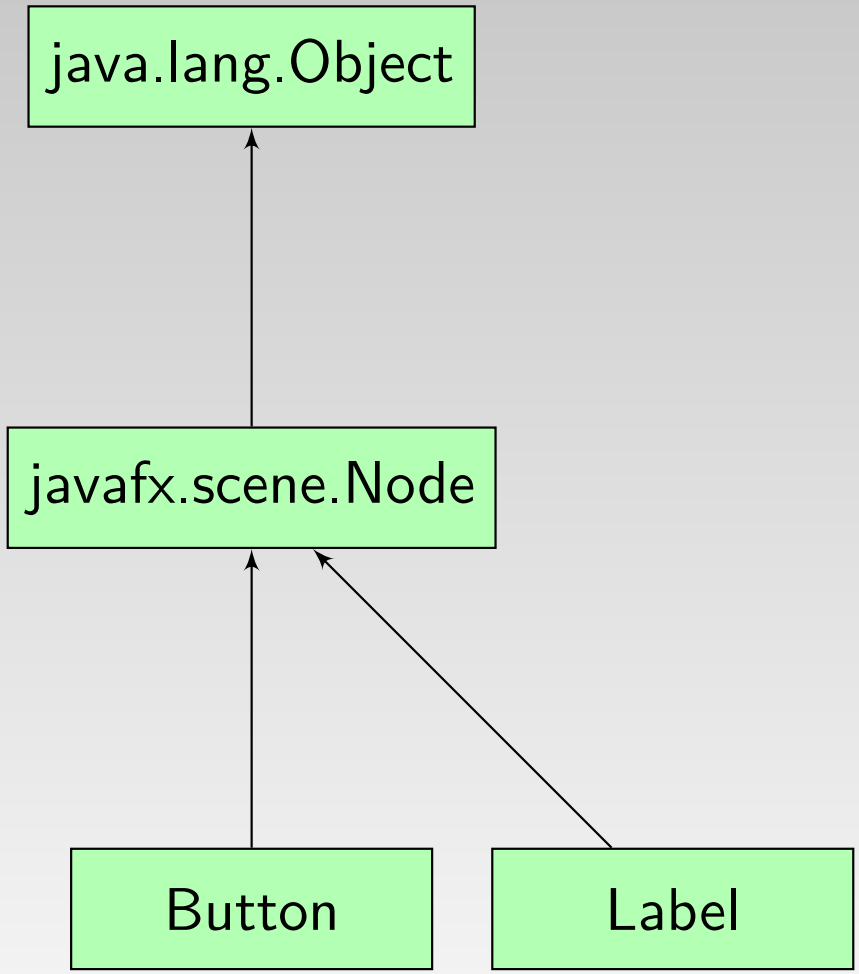
```
1  root.getChildren().add(new ProgressIndicator↩
       (0.6));
2
```



Basic components

```
1  root.getChildren().add(new ColorPicker());
2
```



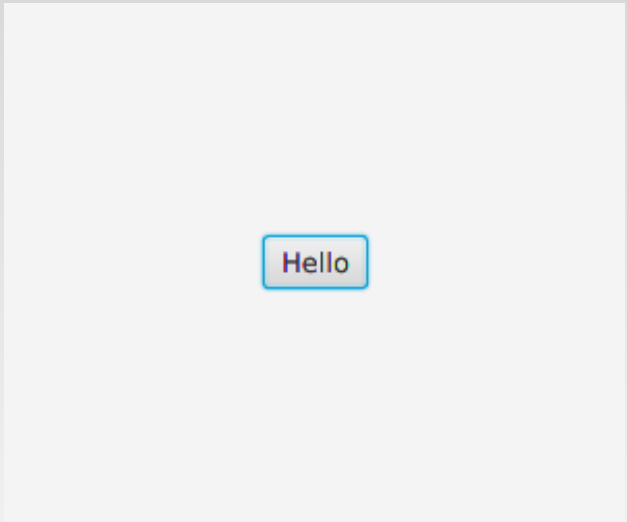

java.lang.Object
javafx.scene.Node
Button
Label

Main menu

```
1  MenuBar menubar = new MenuBar();
2  Menu menu = new Menu("File");
3  MenuItem menuItem = new MenuItem("Open");
4  menu.getItems().add(menuItem);
5  menu.getItems().add(new CheckMenuItem("Check"));
6  menu.getItems().add(new RadioMenuItem("Check"));
7  menu.getItems().add(new SeparatorMenuItem());
8  menubar.getMenus().add(menu);
9  root.getChildren().add(menubar);
```

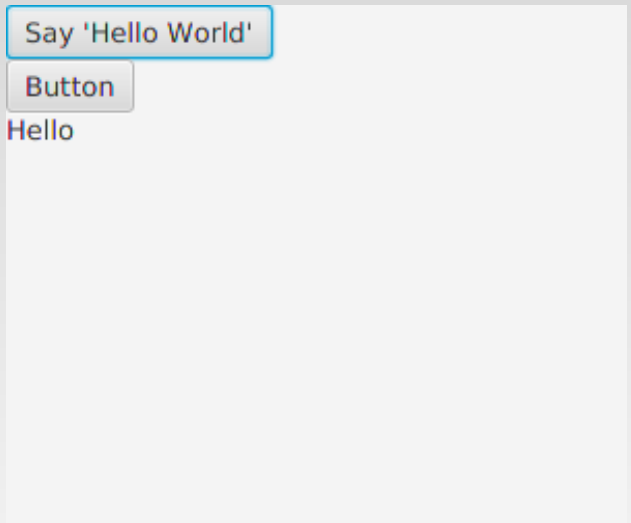
Manu components

```
1   StackPane root = new StackPane();
2   Button btn = new Button();
3   btn.setText("Hello");
4   root.getChildren().add(btn);
5
6   Label lbl = new Label("Hello");
7   root.getChildren().add(lbl);
8
9   Scene scene = new Scene(root, 300, 250);
10
11  primaryStage.setTitle("Hello World!");
12  primaryStage.setScene(scene);
13  primaryStage.show();
14
```



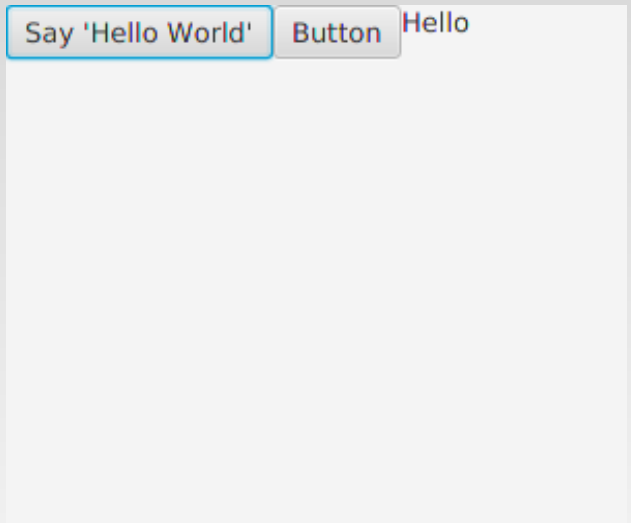
VBox

```
1  VBox root = new VBox();
2
3  Button btn = new Button();
4  Button btn2 = new Button("Button");
5  btn.setText("Say 'Hello World'");
6  Label lbl = new Label("Hello");
7
8  root.getChildren().add(btn);
9  root.getChildren().add(btn2);
10 root.getChildren().add(lbl);
11
12 Scene scene = new Scene(root, 300, 250);
13 primaryStage.setTitle("Hello World!");
14 primaryStage.setScene(scene);
15 primaryStage.show();
16
```



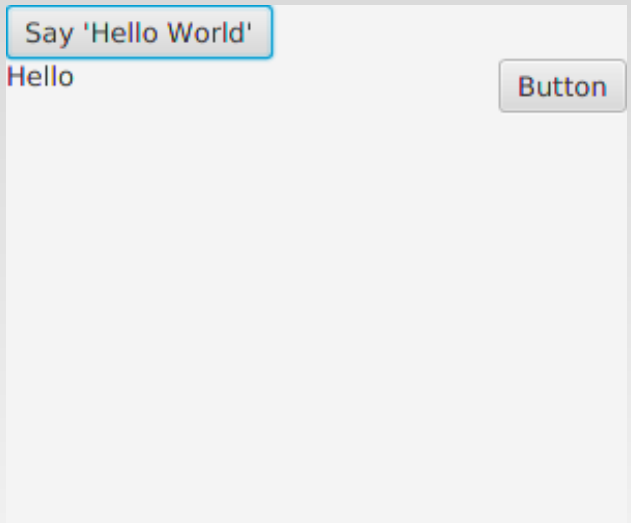
HBox

```
1  HBox root = new HBox();
2
3  Button btn = new Button();
4  Button btn2 = new Button("Button");
5  btn.setText("Say 'Hello World'");
6  Label lbl = new Label("Hello");
7
8  root.getChildren().add(btn);
9  root.getChildren().add(btn2);
10 root.getChildren().add(lbl);
11
12 Scene scene = new Scene(root, 300, 250);
13 primaryStage.setTitle("Hello World!");
14 primaryStage.setScene(scene);
15 primaryStage.show();
16
```



BorderPane

```
1  BorderPane root = new BorderPane();
2
3  Button btn = new Button();
4  Button btn2 = new Button("Button");
5  btn.setText("Say 'Hello World'");
6  Label lbl = new Label("Hello");
7
8  root.setTop(btn);
9  root.setRight(btn2);
10 root.setLeft(lbl);
11
12 Scene scene = new Scene(root, 300, 250);
13 primaryStage.setTitle("Hello World!");
14 primaryStage.setScene(scene);
15 primaryStage.show();
16
```

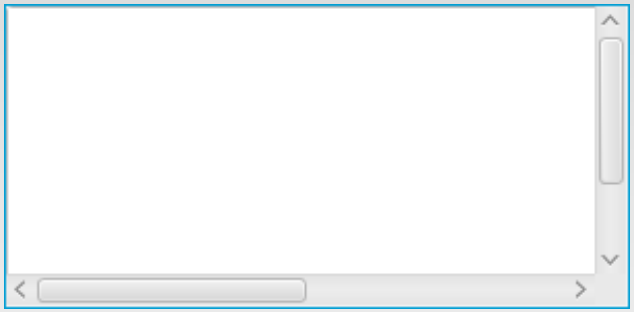


Nesting layout managers

```
1         Pane mainPane = new Pane();
2         Pane pane = new Pane();
3         Button button = new Button();
4
5         pane.getChildren().add(button);
6         mainPane.getChildren().add(pane);
7
8         StackPane root = new StackPane();
9         root.getChildren().add(mainPane);
10        Scene scene = new Scene(root, 300, 250);
11        primaryStage.setTitle("Hello World!");
12        primaryStage.setScene(scene);
13        primaryStage.show();
```

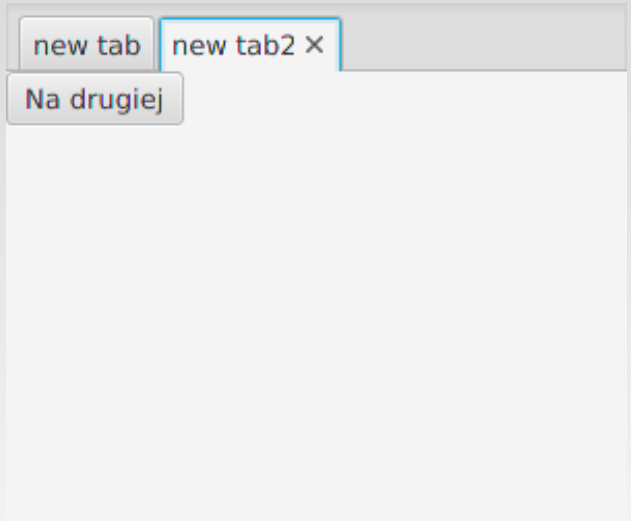
Element scrolling

```
1         ScrollPane sp = new ScrollPane();
2         sp.setContent(new TextArea());
3
4         StackPane root = new StackPane();
5         root.getChildren().add(sp);
6         Scene scene = new Scene(root, 300, ↩
      250);
7         primaryStage.setTitle("Hello World!");
8         primaryStage.setScene(scene);
9         primaryStage.show();
10
```



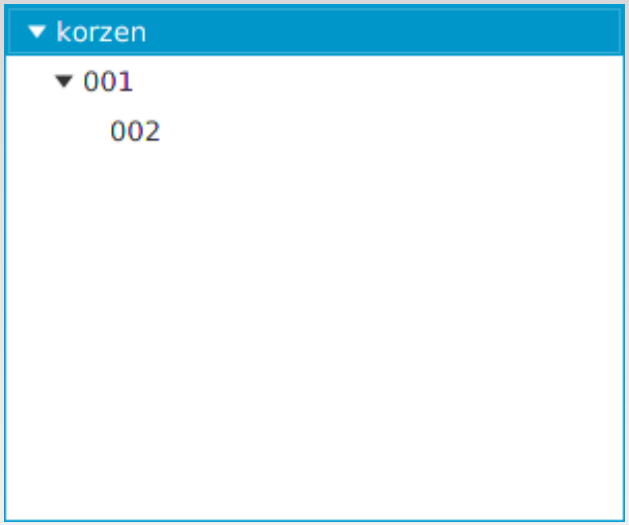
Tabs

```
1         TabPane tabPane = new TabPane();
2         Tab tab = new Tab();
3         tab.setText("new tab");
4         tab.setContent(new Button("Na ↩
      pierwszej"));
5         Tab tab2 = new Tab();
6         tab2.setText("new tab2");
7         tab2.setContent(new Button("Na drugiej↩
      "));
8         tabPane.getTabs().add(tab);
9         tabPane.getTabs().add(tab2);
10
11        StackPane root = new StackPane();
12        root.getChildren().add(tabPane);
13        Scene scene = new Scene(root, 300, ↩
      250);
14        primaryStage.setTitle("Hello World!");
15        primaryStage.setScene(scene);
16        primaryStage.show();
17
```



Tree

```
1  TreeItem<String> treeRoot = new TreeItem("↩
       korzen");
2  TreeItem<String> treeBranch = new TreeItem(↩
       "001");
3  treeRoot.getChildren().add(treeBranch);
4  TreeItem<String> treeLeef = new TreeItem("↩
       002");
5  treeBranch.getChildren().add(treeLeef);
6
7  TreeView<String> tree = new TreeView<>(↩
       treeRoot);
8
9  root.getChildren().add(tree);
10
```



Tables

```
1  public class Element{
2      private final SimpleStringProperty name;
3      private final SimpleStringProperty value;
4      private final SimpleStringProperty anotherValue;
5
6      public Element(String name, String value, String anotherValue) {
7          this.name = new SimpleStringProperty(name);
8          this.value = new SimpleStringProperty(value);
9          this.anotherValue = new SimpleStringProperty(anotherValue);
10     }
11
12     public String getName() {return name.get(); }
13
14     public String getValue() { return value.get();}
15
16     public String getAnotherValue() {
17         return anotherValue.get();
18     }
19
20 }
21
```

Tables

```
1     private TableView table = new TableView();
2
3     final ObservableList<Element> data = FXCollections.observableArrayList(
4             new Element("nazwa", "wartosc", "inna wartosc"),
5             new Element("inna nazwa", "inna wartosc", "bardzo inna")
6     );
7
```

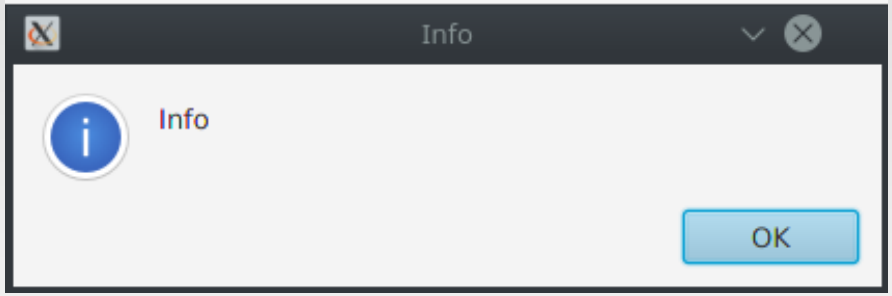
Tables

```
1  TableColumn nameColumn = new TableColumn("↩
       Nazwa");
2  nameColumn.setCellValueFactory(new ↩
       PropertyValueFactory<>("name"));
3  TableColumn valueColumn = new TableColumn("↩
       Wartosc");
4  valueColumn.setCellValueFactory(new ↩
       PropertyValueFactory<>("value"));
5  TableColumn anotherColumn = new TableColumn(↩
       "Inna");
6  anotherColumn.setCellValueFactory(new ↩
       PropertyValueFactory<>("anotherValue"));
7
8  table.getColumns().addAll(nameColumn, ↩
       valueColumn, anotherColumn);
9
10 table.setItems(data);
11
```

Nazwa	Wartosc	Inna
nazwa	wartosc	inna wartosc
inna nazwa	inna wartosc	bardzo inna

Dialogs

```
1  Alert alert = new Alert(AlertType.INFORMATION);
2  alert.setTitle("Info");
3  alert.setHeaderText(null);
4  alert.setContentText("Info");
```

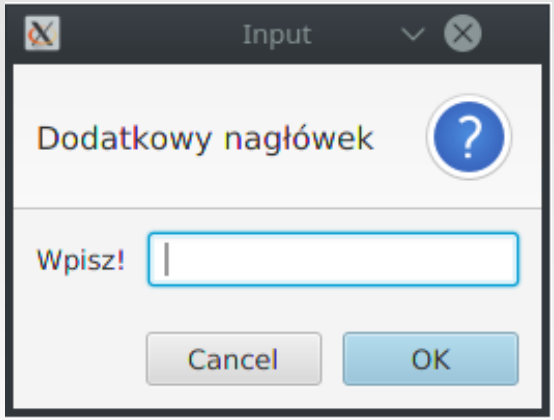


Alert types

- INFORMATION
- WARNING
- CONFIRMATION
- ERROR

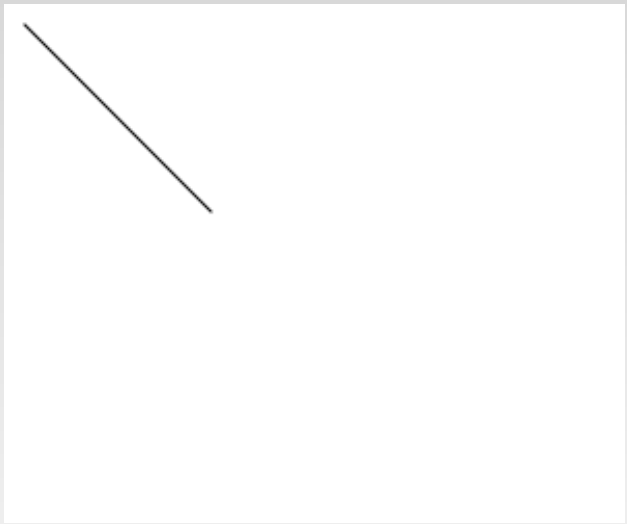
Dialogs

```
1  TextInputDialog dialog = new TextInputDialog();
2
3  dialog.setTitle("Input");
4  dialog.setHeaderText("Dodatkowy naglowek");
5  dialog.setContentText("Wpisz!");
6
7  Optional<String> result = dialog.showAndWait();
```



Drawing

```
1       Group root = new Group();
2       Canvas canvas = new Canvas(300, 250);
3       GraphicsContext gc = canvas.↩
         getGraphicsContext2D();
4       gc.strokeLine(10, 10, 100, 100);
5       root.getChildren().add(canvas);
6       primaryStage.setScene(new Scene(root));
7       primaryStage.show();
8
```



GraphicsContext

- drawImage()
- strokeText()
- strokeRect()
- strokePolygon()
- strokeOval()
- fillRect()

Reacting on events

```
1  public class ActionOwn extends Application implements EventHandler<ActionEvent↩
       > {
2
3      @Override
4      public void start(Stage primaryStage) {
5          StackPane root = new StackPane();
6          Button btn = new Button();
7          btn.setText("Hello");
8          btn.setOnAction(this);
9          root.getChildren().add(btn);
10         ...
11     }
12
13     @Override
14     public void handle(ActionEvent event) {
15         System.out.println("Cliked! :)");
16     }
17
18 }
```

Reacting on events

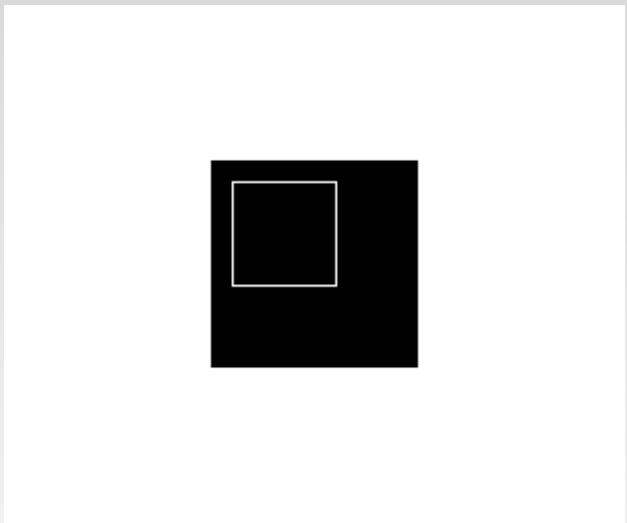
```
1  btn.setOnAction(new EventHandler<ActionEvent>() {
2
3          @Override
4          public void handle(ActionEvent event) {
5              System.out.println("Cliked! :)");
6          }
7      });
```

Reacting on mouse events

```
1         StackPane root = new StackPane();
2         Scene scene = new Scene(root, 300, 250);
3
4         scene.addEventFilter(MouseEvent.MOUSE_CLICKED, new EventHandler<↩
      MouseEvent>() {
5             @Override
6             public void handle(MouseEvent mouseEvent) {
7                 System.out.println("Clicked! " + mouseEvent.getSource());
8             }
9         });
10        primaryStage.setTitle("Hello World!");
11        primaryStage.setScene(scene);
12        primaryStage.show();
```

BufferedImage

```
1  BufferedImage bufferedImage = new ↩
       BufferedImage(100, 100, BufferedImage.↩
       TYPE_INT_RGB);
2  bufferedImage.getGraphics().drawRect(10, 10, ↩
       50, 50);
3
4  Image image = SwingFXUtils.toFXImage(↩
       bufferedImage, null);
5
6  StackPane root = new StackPane();
7  Scene scene = new Scene(root, 300, 250);
8  ImageView imageView = new ImageView();
9  imageView.setImage(image);
10 root.getChildren().add(imageView);
11 primaryStage.setTitle("Hello World!");
12 primaryStage.setScene(scene);
13 primaryStage.show();
14
```

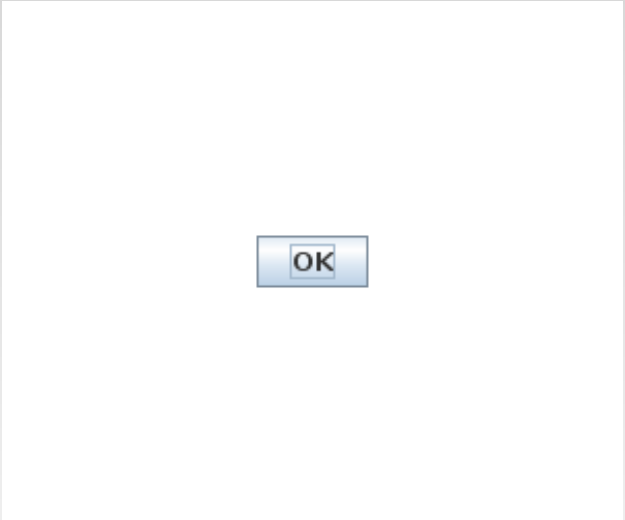


ImageIO

```
1       try {
2         image = ImageIO.read(new File("/home/adam/p.png"));
3         image.getGraphics().drawLine(0, 0, 100, 100);
4         ImageIO.write(image, "png", new File("/home/adam/test.png"));
5       } catch (IOException e) {
6         e.printStackTrace();
7       }
```

BufferedImage

```
1  StackPane root = new StackPane();
2  Scene scene = new Scene(root, 300, 250);
3  SwingNode sn = new SwingNode();
4  sn.setContent(new JButton("OK"));
5  root.getChildren().add(sn);
6  primaryStage.setTitle("Hello World!");
7  primaryStage.setScene(scene);
8  primaryStage.show();
9
```



FXML

- Java FX Markup Language
- Format for describing layouts using XML
- It allows to divide application into tree independent components
  - Models
  - Views
  - Controllers

FXML

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
3  <?import java.lang.*?>
4  <?import java.util.*?>
5  <?import javafx.scene.*?>
6  <?import javafx.scene.control.*?>
7  <?import javafx.scene.layout.*?>
8
9
10 <AnchorPane id="AnchorPane" prefHeight="400.0" prefWidth="600.0" xmlns:fx="↩
       http://javafx.com/fxml/1" xmlns="http://javafx.com/javafx/8" fx:controller="↩
       com.wyklad.gui.javafx.FXMLController">
11    <children>
12       <Button layoutX="14.0" layoutY="14.0" mnemonicParsing="false" text="↩
       Button" onAction="#buttonHandler"/>
13       <TextField layoutX="14.0" layoutY="54.0" text="" fx:id="textField" />
14    </children>
15 </AnchorPane>
```

Controller

```
1  public class FXMLController implements Initializable {
2
3      @FXML
4      private TextField textField;
5
6      /**
7       * Initializes the controller class.
8       */
9      @Override
10     public void initialize(URL url, ResourceBundle rb) {
11         // TODO
12     }
13
14     @FXML
15     protected void buttonHandler(Event e){
16         textField.setText("Hello!");
17     }
18
19 }
```

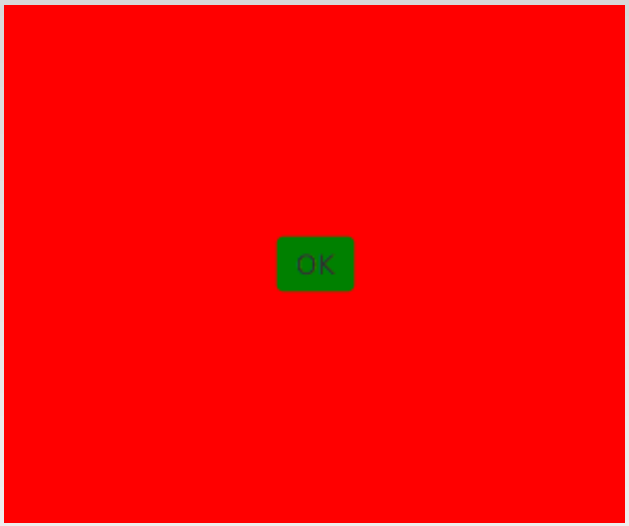
Main class

```
1  public class FXMLWindow extends Application {
2
3      @Override
4      public void start(Stage primaryStage) {
5          try {
6              Pane myPane = (Pane)FXMLLoader.load(FXMLWindow.class.getResource("/↩
       com/wyklad/gui/javafx/FXML.fxml"));
7              Scene myScene = new Scene(myPane);
8              primaryStage.setScene(myScene);
9              primaryStage.show();
10         } catch (IOException ex) {
11             ex.printStackTrace();
12         }
13     }
14
15     public static void main(String[] args) {
16         launch(args);
17     }
18
19 }
```

CSS

```
1  .root{
2      -fx-background-color: red;
3  }
4
5  .button{
6      -fx-background-color: green;
7  }
```

```
1  public class Styles extends Application {
2
3      @Override
4      public void start(Stage primaryStage) {
5
6          Button btn = new Button("OK");
7          StackPane root = new StackPane();
8          root.getChildren().add(btn);
9          Scene scene = new Scene(root, 300, ↩
       250);
10         scene.getStylesheets().add("/com/↩
       wyklad/gui/javafx/style.css");
11         primaryStage.setTitle("Hello World!");
12         primaryStage.setScene(scene);
13         primaryStage.show();
14     }
15
16     public static void main(String[] args) {
17         launch(args);
18     }
19
20 }
21
```



THE END!

Additional reading:

- https://docs.oracle.com/javafx/2/get_started/jfxpub-get_started.htm
- https://docs.oracle.com/javase/8/javase-clienttechnologies.htm
- Thinking in Java chapters by Bruce Eckel (considering Swing library):
  - Graphical User Interfaces