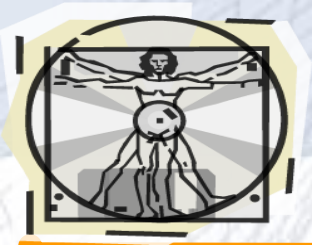
Programing Languages - II
Visual Programming-JavaFX

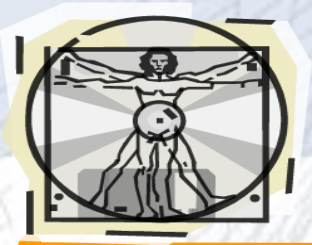
Özgür Koray ŞAHİNGÖZ
Prof.Dr.

Biruni University
Computer Engineering Department



Basic Controls

- Label
- Button
- MenuButton
- SplitMenuButton
- ButtonBar
- ToggleButton
- RadioButton
- CheckBox
- ChoiceBox
- ComboBox
- ListView
- TextField
- PasswordField
- TextArea



Labels

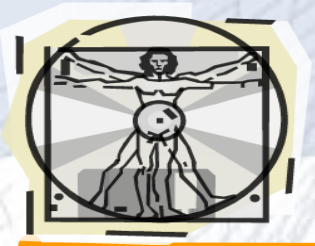
- The JavaFX Label control can display a text or image label inside a JavaFX GUI. The label control must be added to the scene graph to be visible. The JavaFX Label control is represented by the class javafx.scene.control.Label .

Creating a Label

- You create a label control instance by creating an instance of the Label class. Here is a JavaFX Label instantiation example:

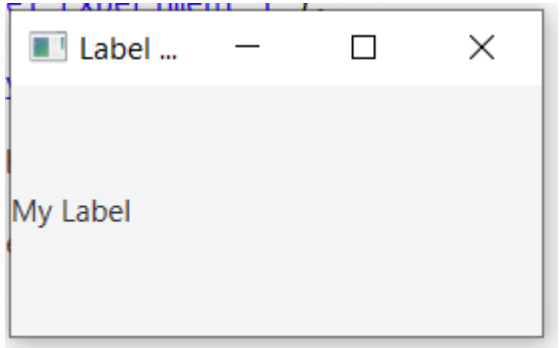
```
Label label = new Label("My Label");
```

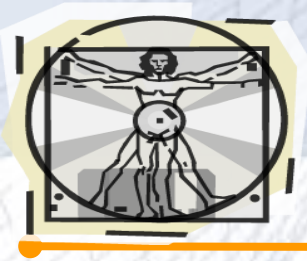
- As you can see, the text to display in the label is passed as parameter to the Label constructor.



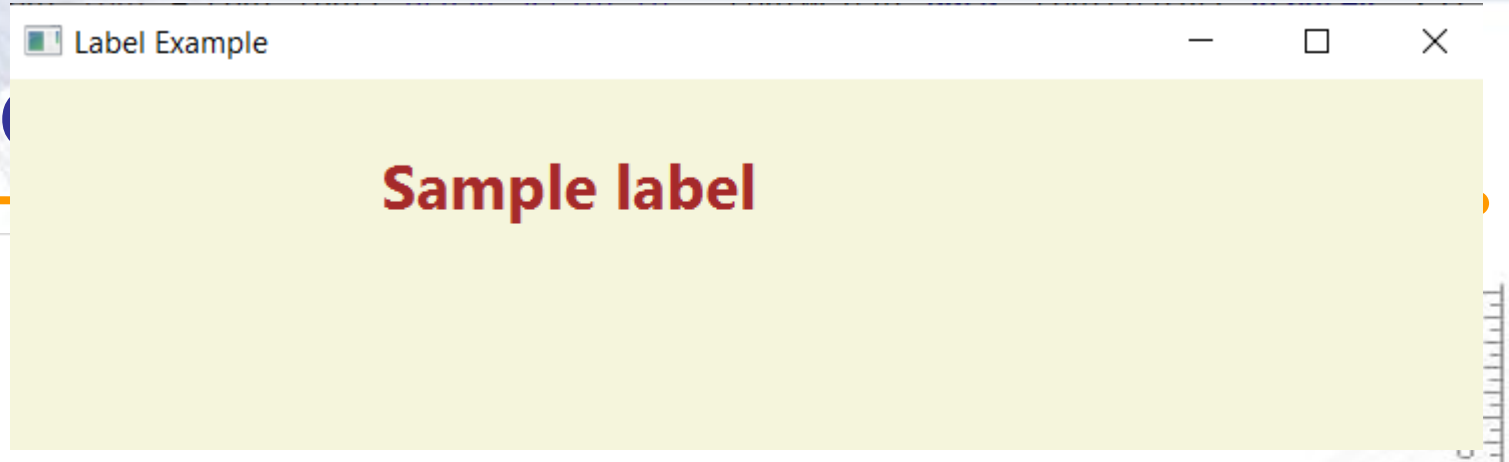
Label Example

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Label;
import javafx.stage.Stage;

public class Main extends Application  {
public void start(Stage primaryStage) throws Exception {
        primaryStage.setTitle("Label Experiment 1");

        Label label = new Label("My Label");

        Scene scene = new Scene(label, 200, 100);

        primaryStage.setScene(scene);
        primaryStage.show();
    }
    public static void main(String[] args) {
        Application.launch(args);
    }
}
```

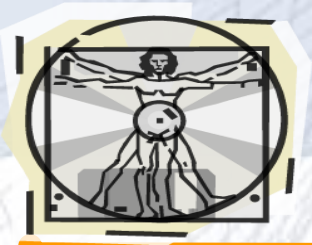




Label Example




```
1  package application;
2  import javafx.application.Application;
11 public class Main extends Application {
12    public void start(Stage stage) {
13       //Creating a Label
14       Label label = new Label("Sample label");
15       //Setting font to the label
16       Font font = Font.font("Brush Script MT", FontWeight.BOLD, FontPosture.REGULAR, 25);
17       label.setFont(font);
18       //Filling color to the label
19       label.setTextFill(Color.BROWN);
20       //Setting the position
21       label.setTranslateX(150);
22       label.setTranslateY(25);
23       Group root = new Group();
24       root.getChildren().add(label);
25       //Setting the stage
26       Scene scene = new Scene(root, 595, 150, Color.BEIGE);
27       stage.setTitle("Label Example");
28       stage.setScene(scene);
29       stage.show();
30    }
31    public static void main(String args[]){
32       launch(args);
33    }
34 }
```



Button

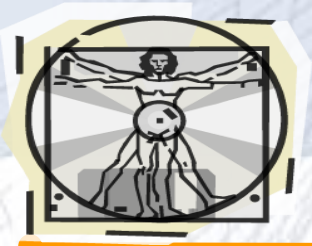
- A JavaFX Button control enables a JavaFX application to have some action executed when the application user clicks the button. The JavaFX Button control is represented by the class javafx.scene.control.Button . A JavaFX Button can have a text and an icon on it which indicate to the user what clicking the button will do.

Creating a Button

- You create a button control by creating an instance of the Button class. Here is a JavaFX Button instantiation example:

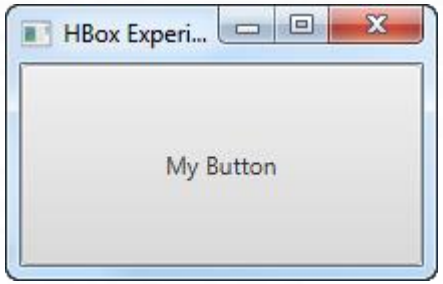
```
Button button = new Button("My Label");
```

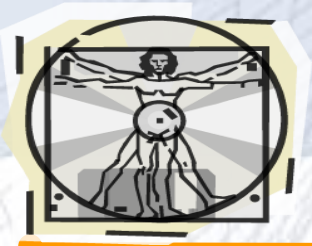
- The text to be displayed on the button is passed as parameters to the Button constructor.



Button Example

```
1  package application;
2  import javafx.application.Application;
3  import javafx.scene.Scene;
4  import javafx.scene.control.Button;
5  import javafx.scene.control.Label;
6  import javafx.stage.Stage;
7
8
9  public class ButtonExperiments extends Application  {
10
11     @Override
12     public void start(Stage primaryStage) throws Exception {
13         primaryStage.setTitle("HBox Experiment 1");
14
15         Button button = new Button("My Button");
16
17         Scene scene = new Scene(button, 200, 100);
18         primaryStage.setScene(scene);
19         primaryStage.show();
20
21     }
22
23     public static void main(String[] args) {
24         Application.launch(args);
25     }
26 }
```





Button events

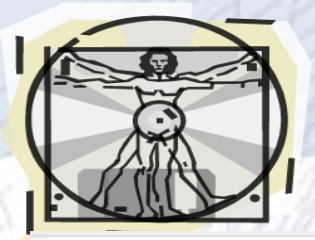
In order to respond to the click of a button you need to attach an event listener to the `Button` object. Here is how that looks:

```
button.setOnAction(new EventHandler() {
    @Override
    public void handle(ActionEvent actionEvent) {
        //... do something in here.
    }
});
```

Here is how attaching a click event listener looks with a **Java Lambda expression**:

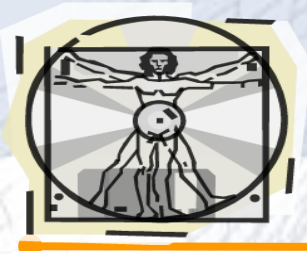
```
button.setOnAction(actionEvent ->  {
    //... do something in here.
});
```

Finally, let us see a full example that changes the text of a **JavaFX Label** when the button is clicked:



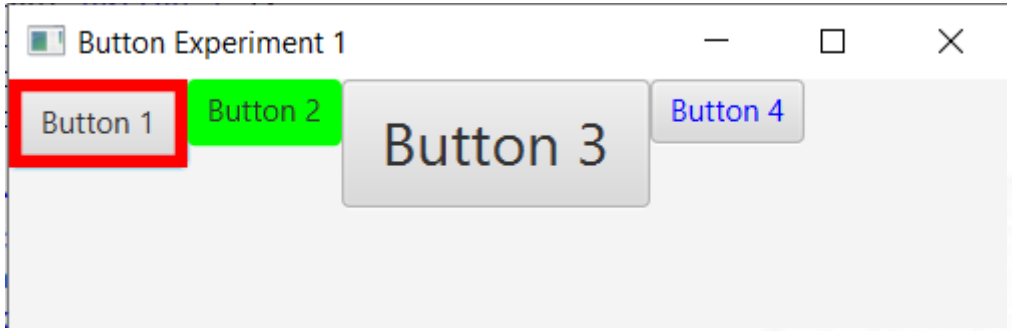
Actions-Events

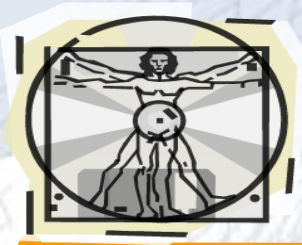
```
1  package application;
2  import javafx.application.Application;
10
11 public class Main extends Application  {
12
13
14     @Override
15     public void start(Stage primaryStage) throws Exception {
16         primaryStage.setTitle("HBox Experiment 1");
17
18         Label label = new Label("Not clicked");
19         Button button = new Button("Click");
20
21         button.setOnAction(value ->  {
22            label.setText("Clicked!");
23         });
24
25         HBox hbox = new HBox(button, label);
26
27         Scene scene = new Scene(hbox, 200, 100);
28         primaryStage.setScene(scene);
29         primaryStage.show();
30
31     }
32
33     public static void main(String[] args) {
34         Application.launch(args);
35     }
36 }
```



Designing Different Buttons

```
1  package application;
2  import javafx.application.Application;
7
8  public class Main extends Application  {
9      @Override
10     public void start(Stage primaryStage) throws Exception {
11         primaryStage.setTitle("Button Experiment 1");
12
13         Button button1 = new Button("Button 1");
14         Button button2 = new Button("Button 2");
15         Button button3 = new Button("Button 3");
16         Button button4 = new Button("Button 4");
17
18         button1.setStyle("-fx-border-color: #ff0000; -fx-border-width: 5px;");
19         button2.setStyle("-fx-background-color: #00ff00");
20         button3.setStyle("-fx-font-size: 2em; ");
21         button4.setStyle("-fx-text-fill: #0000ff");
22
23         HBox hbox = new HBox(button1, button2, button3, button4);
24
25         Scene scene = new Scene(hbox, 400, 100);
26         primaryStage.setScene(scene);
27         primaryStage.show();
28     }
29
30     public static void main(String[] args) {
31         Application.launch(args);
32     }
33 }
```

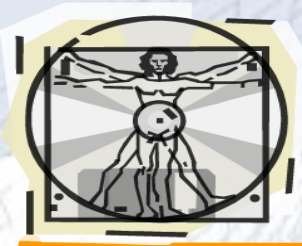




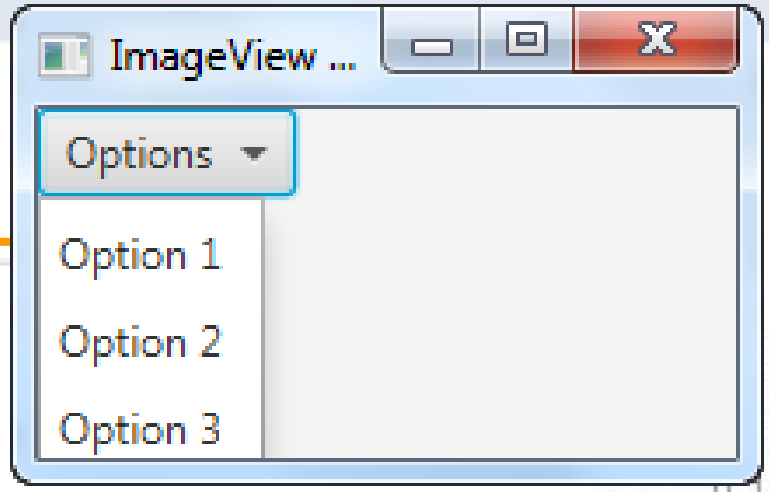
MenuButton

Creating a MenuButton

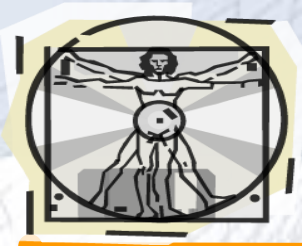
- You create a JavaFX MenuButton by creating an instance of the MenuButton class. The MenuButton constructor takes a button text and a button graphic. You can pass null for the text and / or the graphic, in case you want a MenuButton without either text or graphic. Here is an example of creating a JavaFX MenuButton with only a text label:
  - MenuItem menuItem1 = new MenuItem("Option 1");
  - MenuItem menuItem2 = new MenuItem("Option 2");
  - MenuItem menuItem3 = new MenuItem("Option 3");
  - MenuButton menuButton = new MenuButton("Options", null, menuItem1, menuItem2, menuItem3);
- First 3 MenuItem instances are created, each with a different text. Then a MenuButton instance is created, passing a button text, a graphic icon (null) and the 3 MenuItem instances as parameter to the MenuButton constructor.



MenuButton Example

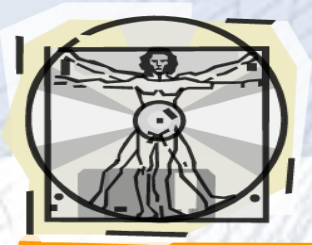



```
1  package application;
2  import javafx.application.Application;
10 public class Main extends Application  {
11     @Override
12     public void start(Stage primaryStage) throws Exception {
13         primaryStage.setTitle("ImageView Experiment 1");
14
15         MenuItem menuItem1 = new MenuItem("Option 1");
16         MenuItem menuItem2 = new MenuItem("Option 2");
17         MenuItem menuItem3 = new MenuItem("Option 3");
18
19         MenuButton menuButton = new MenuButton("Options", null, menuItem1, menuItem2, menuItem3);
20
21         HBox hbox = new HBox(menuButton);
22
23         Scene scene = new Scene(hbox, 200, 100);
24         primaryStage.setScene(scene);
25         primaryStage.show();
26     }
27     public static void main(String[] args) {
28         Application.launch(args);
29     }
30 }
```



Adding Actions

```
    public void start(Stage primaryStage) throws Exception {
        primaryStage.setTitle("ImageView Experiment 1");

        MenuItem menuItem1 = new MenuItem("Option 1");
        MenuItem menuItem2 = new MenuItem("Option 2");
        menuItem2.setOnAction(new EventHandler<ActionEvent>() {
            @Override
            public void handle(ActionEvent event) {
                System.out.println("Option 2 selected");
            }
        });
        MenuItem menuItem3 = new MenuItem("Option 3");

        menuItem3.setOnAction(new EventHandler<ActionEvent>() {
            @Override
            public void handle(ActionEvent event) {
                System.out.println("Option 3 selected");
            }
        });

        MenuButton menuButton = new MenuButton("Options", null, menuItem1, menuItem2, menuItem3);
```



RadioButton

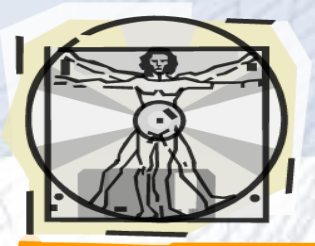
- A JavaFX RadioButton is a button that can be selected or not selected. The RadioButton is very similar to the JavaFX ToggleButton, but with the difference that a RadioButton cannot be "unselected" once selected. If RadioButtons are part of a ToggleGroup then once a RadioButton has been selected for the first time, there must be one RadioButton selected in the ToggleGroup .
- The JavaFX RadioButton is represented by the class javafx.scene.control.RadioButton. The RadioButton class is a subclass of the ToggleButton class.

Creating a RadioButton

- You create a JavaFX RadioButton using its constructor. Here is a JavaFX RadioButton instantiation example:

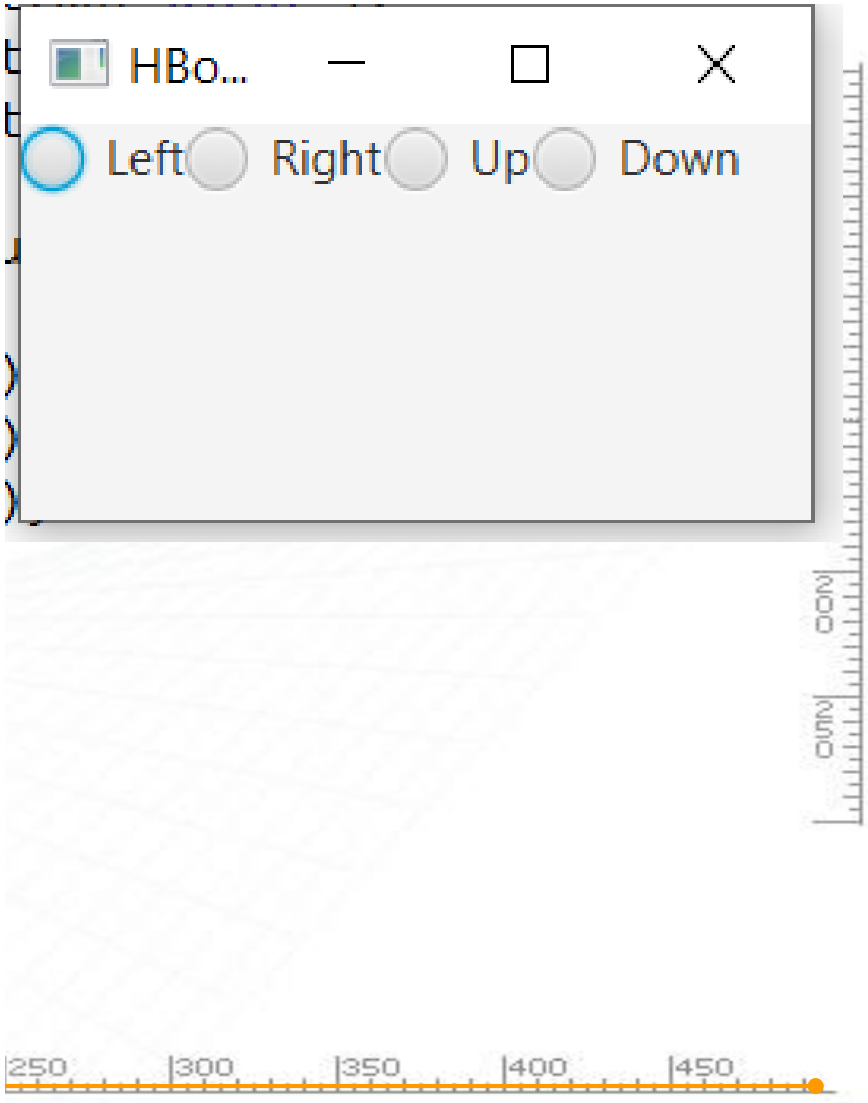
```
RadioButton radioButton1 = new RadioButton("Left");
```

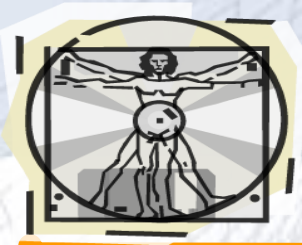
- The String passed as parameter to the RadioButton constructor is displayed next to the RadioButton.



RadioButton Example

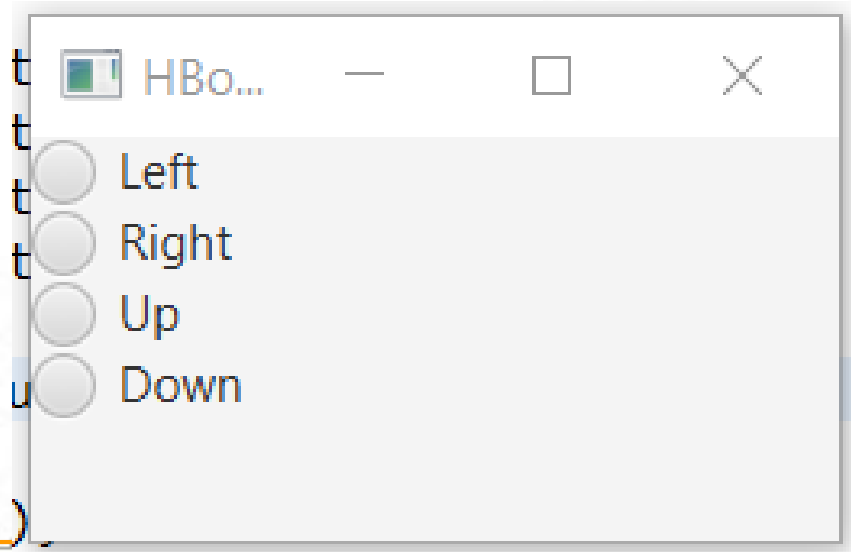
```
1  package application;
2  import javafx.application.Application;
8
9  public class Main extends Application {
10     public void start(Stage primaryStage) throws Exception {
11         primaryStage.setTitle("HBox Experiment 1");
12
13         RadioButton radioButton1 = new RadioButton("Left");
14         RadioButton radioButton2 = new RadioButton("Right");
15         RadioButton radioButton3 = new RadioButton("Up");
16         RadioButton radioButton4 = new RadioButton("Down");
17
18         ToggleGroup radioGroup = new ToggleGroup();
19
20         radioButton1.setToggleGroup(radioGroup);
21         radioButton2.setToggleGroup(radioGroup);
22         radioButton3.setToggleGroup(radioGroup);
23         radioButton4.setToggleGroup(radioGroup);
24
25         HBox hbox = new HBox(radioButton1, radioButton2, radioButton3, radioButton4);
26
27         Scene scene = new Scene(hbox, 200, 100);
28         primaryStage.setScene(scene);
29         primaryStage.show();
30     }
31     public static void main(String[] args) {
32         Application.launch(args);
33     }
34 }
```

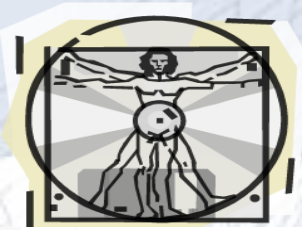




Vertical Layout

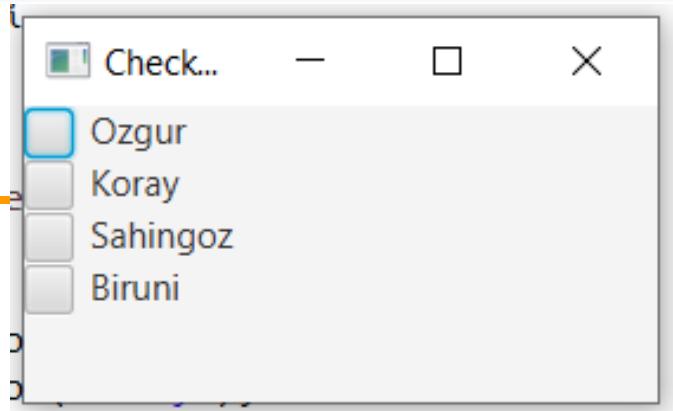
```
        ToggleGroup radioGroup = new ToggleGroup();

        radioButton1.setToggleGroup(radioGroup);
        radioButton2.setToggleGroup(radioGroup);
        radioButton3.setToggleGroup(radioGroup);
        radioButton4.setToggleGroup(radioGroup);

        VBox hbox = new VBox(radioButton1, radioButton2, radioButton3, radioButton4);

        Scene scene = new Scene(hbox, 200, 100);
        primaryStage.setScene(scene);
        primaryStage.show();

    }
```

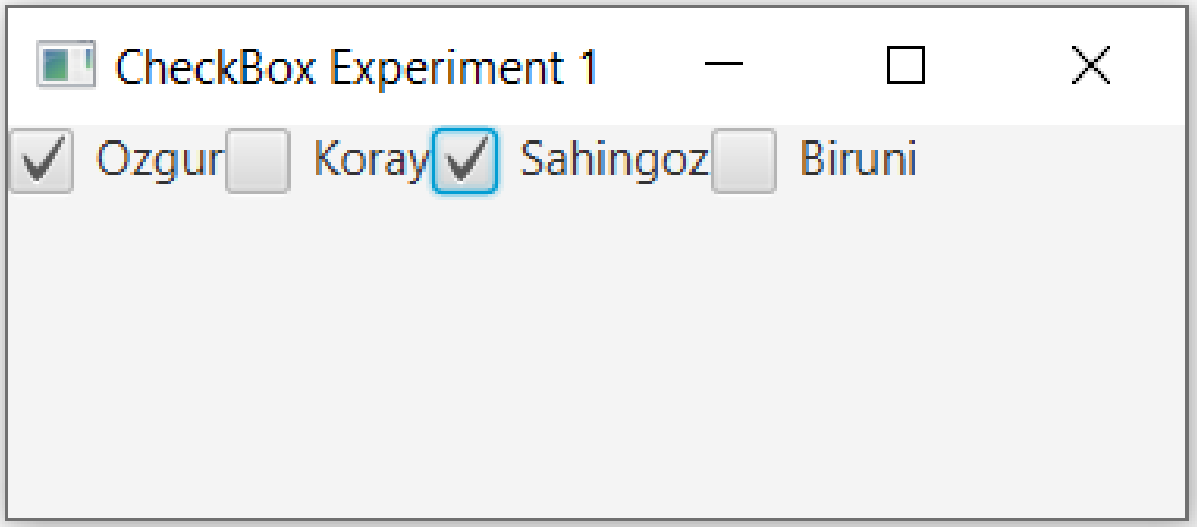


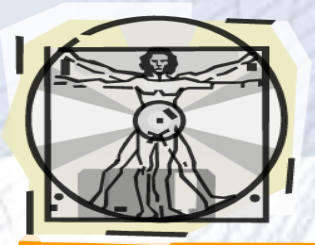


CheckBox

```
1  package application;
2  import javafx.application.Application;
7
8
9  public class Main extends Application  {
10
11
12     @Override
13     public void start(Stage primaryStage) throws Exception {
14         primaryStage.setTitle("CheckBox Experiment 1");
15
16         CheckBox checkBox1 = new CheckBox("Ozgur");
17         CheckBox checkBox2 = new CheckBox("Koray");
18         CheckBox checkBox3 = new CheckBox("Sahingoz");
19         CheckBox checkBox4 = new CheckBox("Biruni");
20
21         HBox hbox = new HBox(checkBox1,checkBox2,checkBox3,checkBo
22
23         Scene scene = new Scene(hbox, 300, 100);
24         primaryStage.setScene(scene);
25         primaryStage.show();
26
27     }
28
29     public static void main(String[] args) {
30         Application.launch(args);
31     }
32
33 }
```

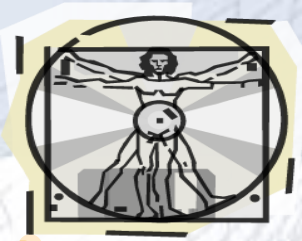






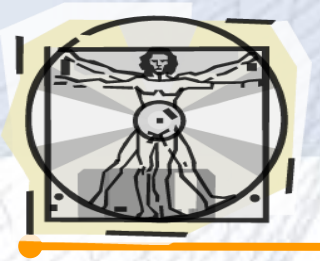
Adding Action Event

```
@Override
public void start(Stage primaryStage) throws Exception {
    primaryStage.setTitle("CheckBox Experiment 1");

    CheckBox checkBox1 = new CheckBox("Ozgur");
    CheckBox checkBox2 = new CheckBox("Koray");
    CheckBox checkBox3 = new CheckBox("Sahingoz");
    CheckBox checkBox4 = new CheckBox("Biruni");
    checkBox4.setOnAction(new EventHandler<ActionEvent>() {
        @Override
        public void handle(ActionEvent event) {
            System.out.println("Biruni CheckBox selected");
        }
    });

    VBox hbox = new VBox(checkBox1,checkBox2,checkBox3,checkBox4);
```



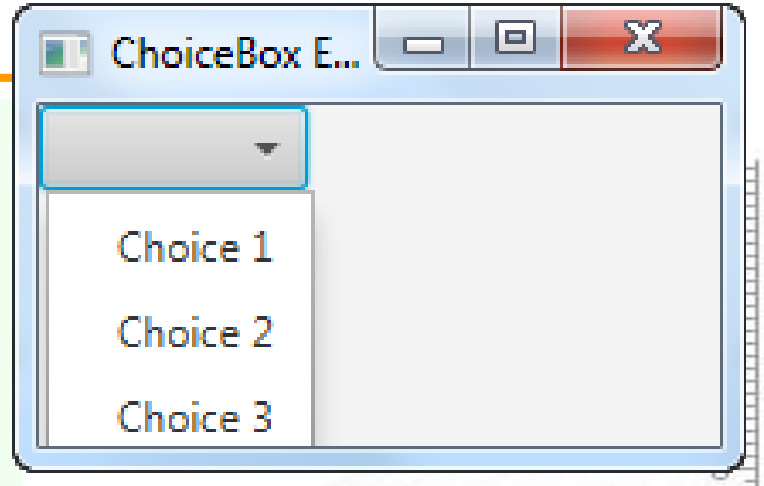
ChoiceBox

- The JavaFX ChoiceBox control enables users to choose an option from a predefined list of choices. The JavaFX ChoiceBox control is represented by the class javafx.scene.control.ChoiceBox . This JavaFX ChoiceBox tutorial will explain how to use the ChoiceBox class.



ChoiceBox

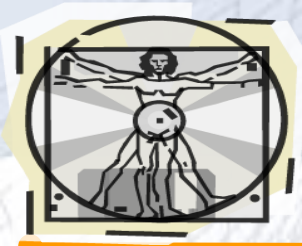
```
public class ChoiceBoxExperiments extends Application  {

    @Override
    public void start(Stage primaryStage) throws Exception {
        primaryStage.setTitle("ChoiceBox Experiment 1");

        ChoiceBox choiceBox = new ChoiceBox();

        choiceBox.getItems().add("Choice 1");
        choiceBox.getItems().add("Choice 2");
        choiceBox.getItems().add("Choice 3");

        HBox hbox = new HBox(choiceBox);

        Scene scene = new Scene(hbox, 200, 100);
        primaryStage.setScene(scene);
        primaryStage.show();

    }

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



Reading the Selected Value

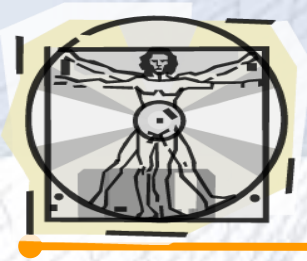
You can read the selected value of a `ChoiceBox` via its `getValue()` metho
an example of calling `getValue()`:

```
String value = (String) choiceBox.getValue();
```



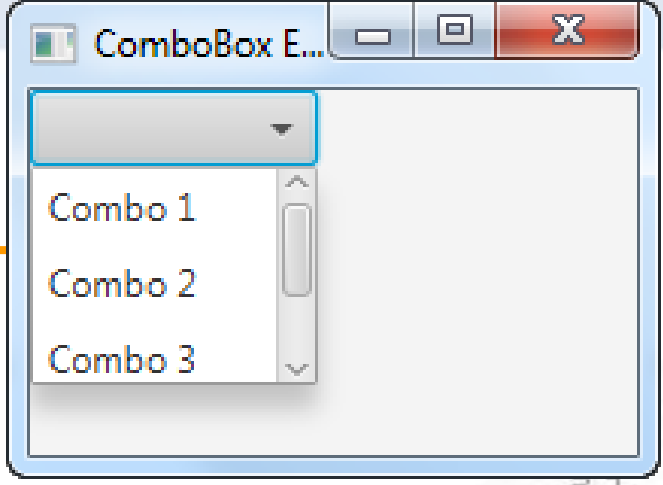
ComboBox

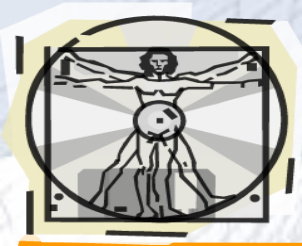
- The JavaFX ComboBox control enables users to choose an option from a predefined list of choices, or type in another value if none of the predefined choices matches what the user want to select. The JavaFX ComboBox control is represented by the class javafx.scene.control.ComboBox . This JavaFX ComboBox tutorial will explain how to use the ComboBox class.



ComboBox Example

```
public class ComboBoxExperiments extends Application  {


    @Override
    public void start(Stage primaryStage) throws Exception {
        primaryStage.setTitle("ComboBox Experiment 1");

        ComboBox comboBox = new ComboBox();

        comboBox.getItems().add("Choice 1");
        comboBox.getItems().add("Choice 2");
        comboBox.getItems().add("Choice 3");


        HBox hbox = new HBox(comboBox);

        Scene scene = new Scene(hbox, 200, 120);
        primaryStage.setScene(scene);
        primaryStage.show();

    }

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



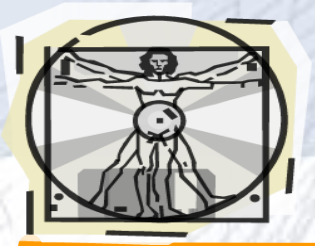


ComboBox Example

```
14      public void start(Stage primaryStage) throws Exception {
15          primaryStage.setTitle("ComboBox Experiment 1");
16
17          ComboBox comboBox = new ComboBox();
18
19          comboBox.getItems().add("Choice 1");
20          comboBox.getItems().add("Choice 2");
21          comboBox.getItems().add("Choice 3");
22
23          comboBox.setOnAction((event) -> {
24              int selectedIndex = comboBox.getSelectionModel().getSelectedIndex();
25              Object selectedItem = comboBox.getSelectionModel().getSelectedItem();
26
27              System.out.println("Selection made: [" + selectedIndex + "] " + selectedItem);
28              System.out.println("   ComboBox.getValue(): " + comboBox.getValue());
29          });
30
31          HBox hbox = new HBox(comboBox);
32
33          Scene scene = new Scene(hbox, 200, 120);
34          primaryStage.setScene(scene);
35          primaryStage.show();
```

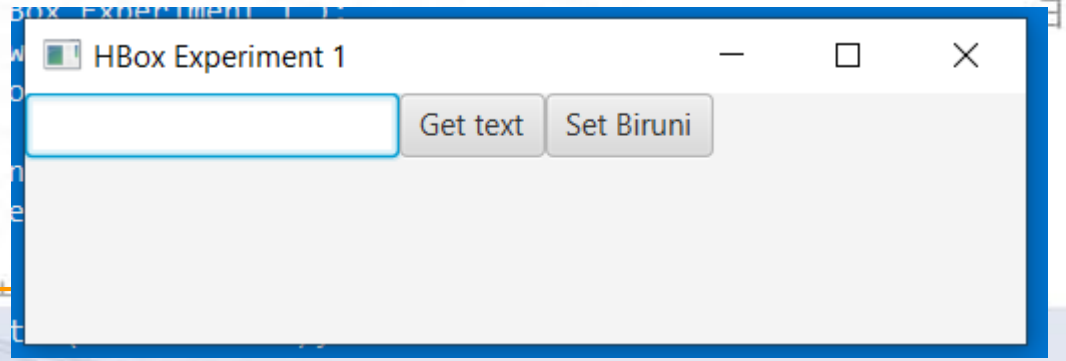
Problems | Javadoc | Declaration | Console

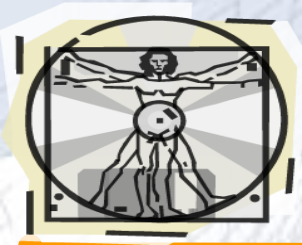
```
<terminated> Main (1) [Java Application] C:\Users\osahingoz\.p2\pool\pl
Selection made: [1] Choice 2
   ComboBox.getValue(): Choice 2
Selection made: [2] Choice 3
   ComboBox.getValue(): Choice 3
```



TextField

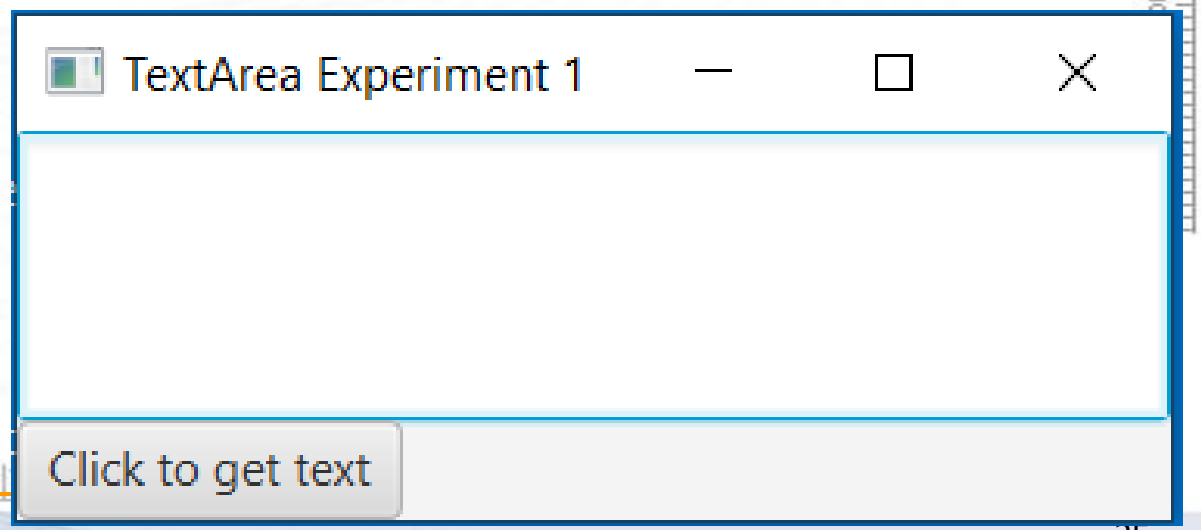
```
public class Main extends Application  {
    @Override
    public void start(Stage primaryStage) throws Exception {
        primaryStage.setTitle("HBox Experiment 1");
        TextField textField = new TextField();
        Button button = new Button("Get text");

        button.setOnAction(action -> {
            System.out.println(textField.getText());
        });

        Button button2 = new Button("Set Biruni");
        button2.setOnAction(action -> {
            textField.setText("Biruni");
        });

        HBox hbox = new HBox(textField, button, button2);

        Scene scene = new Scene(hbox, 400, 100);
        primaryStage.setScene(scene);
        primaryStage.show();
    }
```

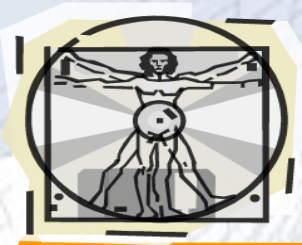




TextField Example

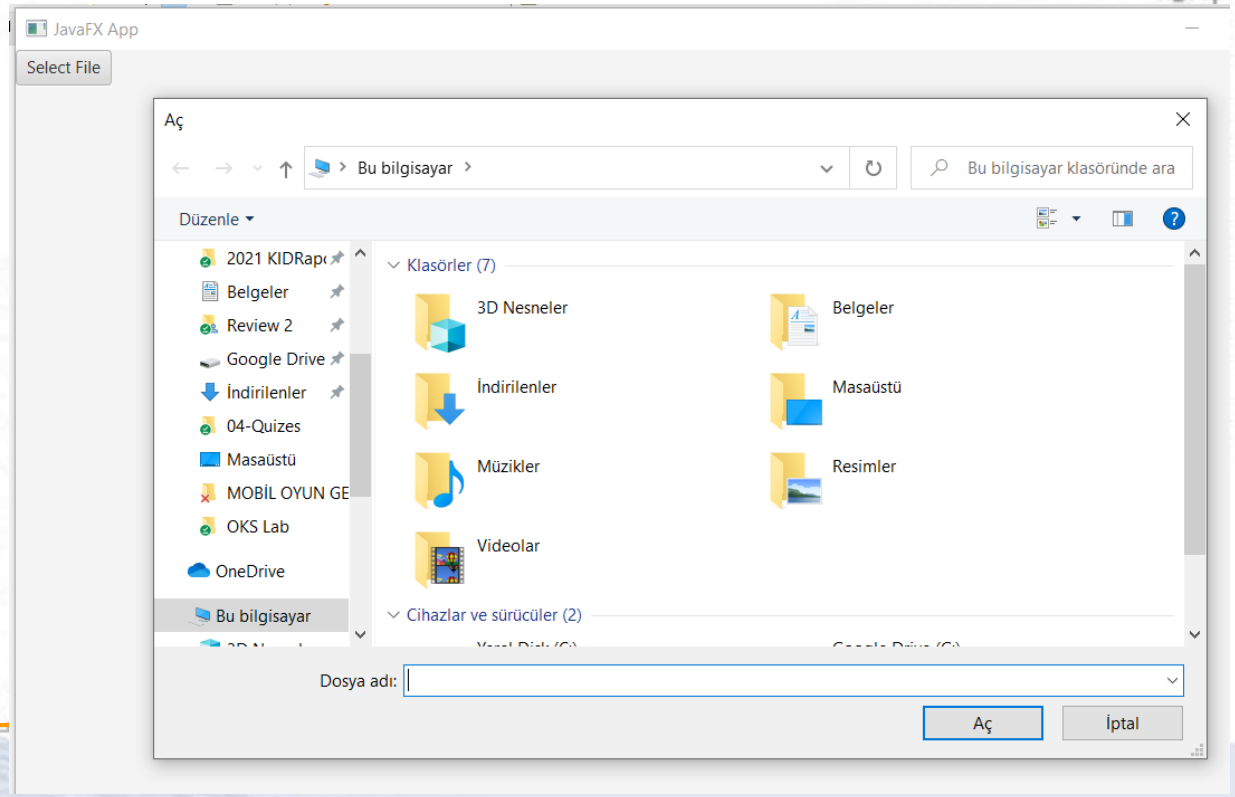
```
public class Main extends Application  {
    @Override
    public void start(Stage primaryStage) throws Exception {
        primaryStage.setTitle("TextArea Experiment 1");
        TextArea textArea = new TextArea();
        Button button = new Button("Click to get text");
        button.setMinWidth(50);

        button.setOnAction(action -> {
            System.out.println(textArea.getText());
            textArea.setText("Clicked!");
        });
        VBox vbox = new VBox(textArea, button);
        Scene scene = new Scene(vbox, 200, 100);
        primaryStage.setScene(scene);
        primaryStage.show();
    }
    public static void main(String[] args) {
        Application.launch(args);
    }
}
```

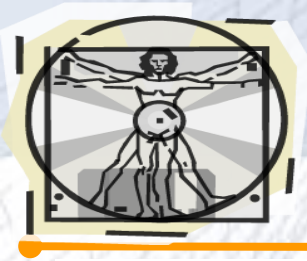




FileChooser

```
public void start(Stage primaryStage) {
        primaryStage.setTitle("JavaFX App");
        FileChooser fileChooser = new FileChooser();
        Button button = new Button("Select File");
        button.setOnAction(e -> {
            File selectedFile = fileChooser.showOpenDialog(primaryStage);
        });

        VBox vBox = new VBox(button);
        Scene scene = new Scene(vBox, 960, 600);

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





FileChooser Example

Setting Initial Directory

You can set the initial directory displayed in the JavaFX `FileChooser` via its `setInitialDirectory()` method. Here is an example of setting the initial directory of a `FileChooser` dialog:

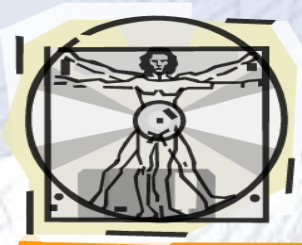
```
fileChooser.setInitialDirectory(new File("data"));
```

This example sets the initial directory displayed by the `FileChooser` to `data`.

Setting Initial File Name

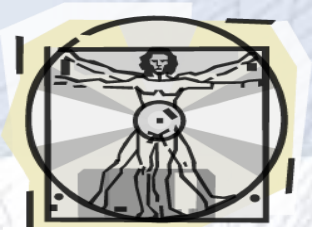
You can set the initial file name to display in the `FileChooser` . Some platforms (e.g. Windows) may ignore this setting, though. Here is an example of setting the initial file name of a `FileChooser`:

```
fileChooser.setInitialFileName("myfile.txt");
```



Use Of Menu

```
public void start(Stage primaryStage) {
        primaryStage.setTitle("JavaFX App");

        MenuBar menuBar = new MenuBar();
        Menu menu = new Menu("Menu 1");
        MenuItem menuItem1 = new MenuItem("Item 1");
        MenuItem menuItem2 = new MenuItem("Item 2");

        menu.getItems().add(menuItem1);
        menu.getItems().add(menuItem2);
        menuBar.getMenus().add(menu);

        Menu menu2 = new Menu("Menu 2");
        menuBar.getMenus().add(menu2);

        VBox vBox = new VBox(menuBar);

        Scene scene = new Scene(vBox, 960, 600);
        primaryStage.setScene(scene);
        primaryStage.show();
    }
```



Filters

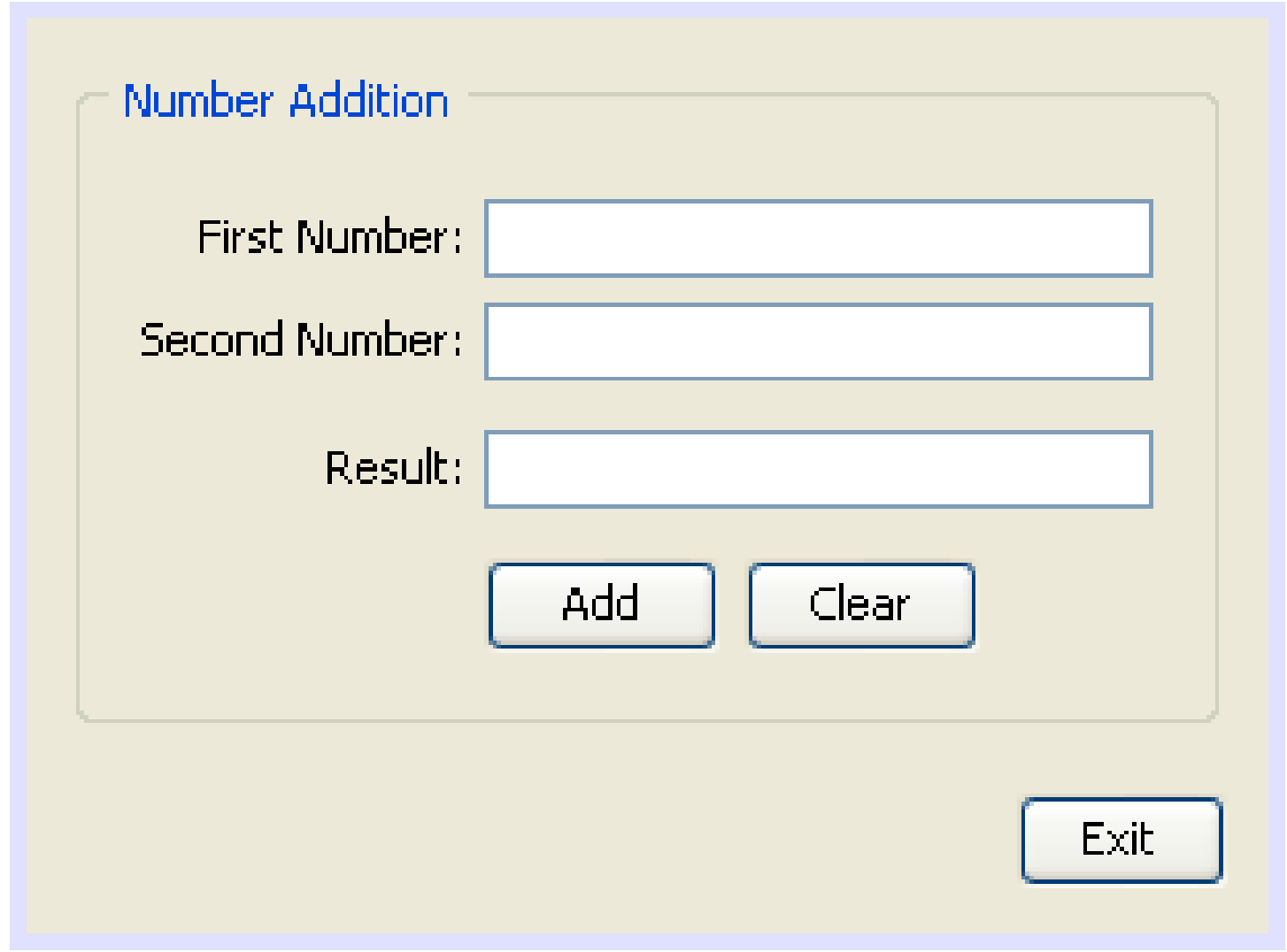
Adding File Name Filters

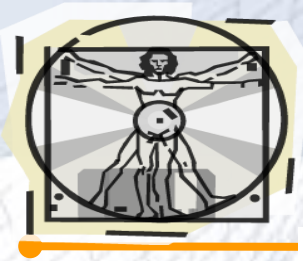
It is possible to add file name filters to a JavaFX `FileChooser`. File name filters are used to filter out what files are shown in the `FileChooser` when the user browses around the file system. Here is an example of adding file name filters:

```
FileChooser fileChooser = new FileChooser();

fileChooser.getExtensionFilters().addAll(
     new FileChooser.ExtensionFilter("Text Files", "*.txt")
    ,new FileChooser.ExtensionFilter("HTML Files", "*.htm")
);
```

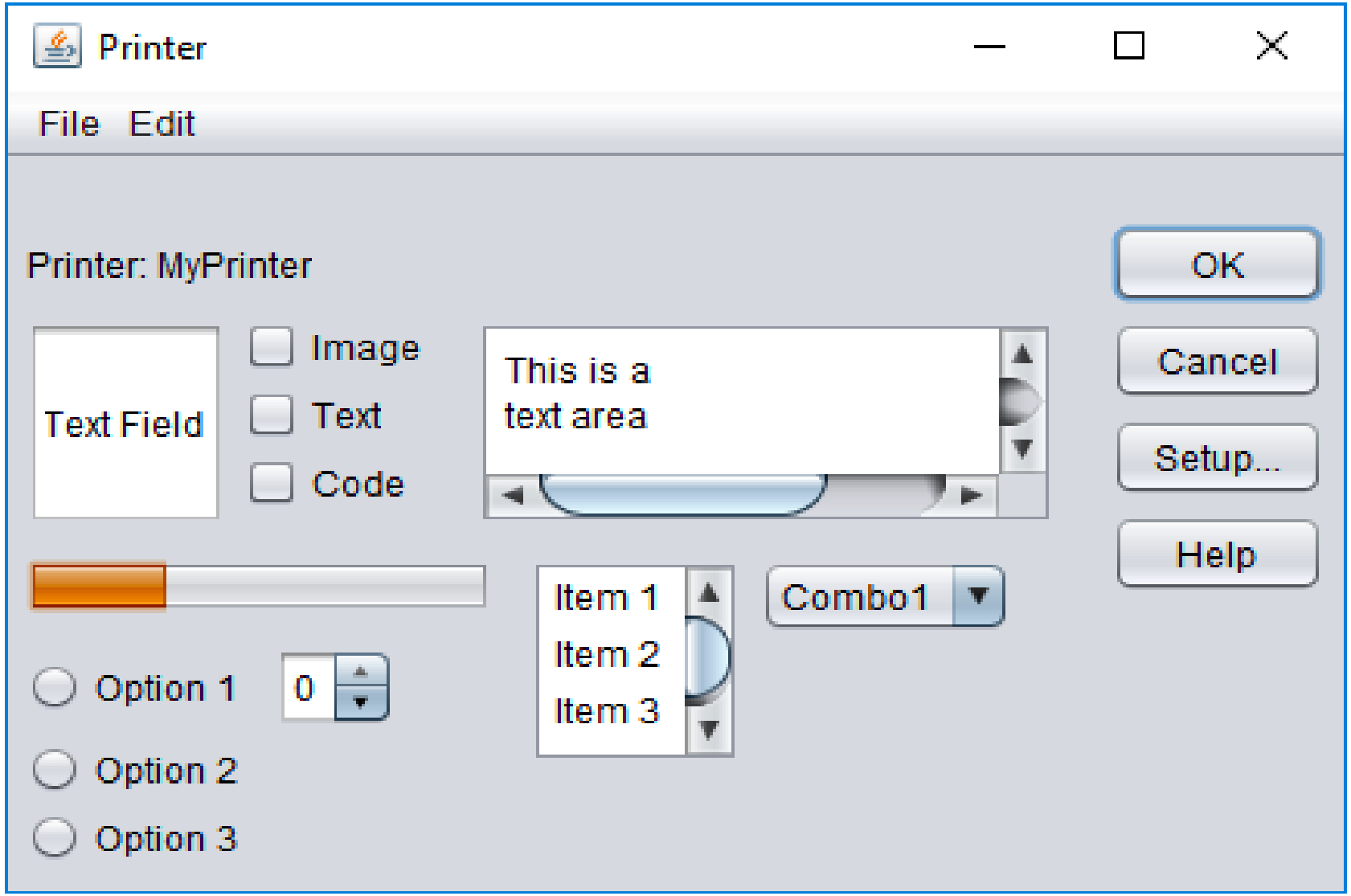
This examples adds two file name filters to the `FileChooser`. The user can choose between these file name filters inside the `FileChooser` dialog.

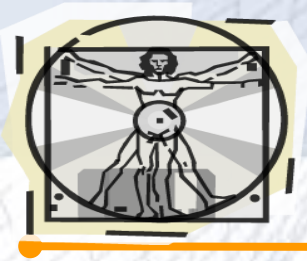
GUI Example-1





GUI Example-2





GUI Example-3

Name

First Name: Last Name:

Title: Nickname:

Format: Item 1

Item 1
Item 2
Item 3
Item 4

E-mail

E-mail Address:

Add

Edit

Remove

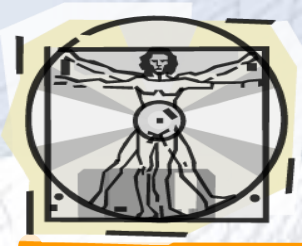
As Default

Item 1
Item 2
Item 3
Item 4
Item 5

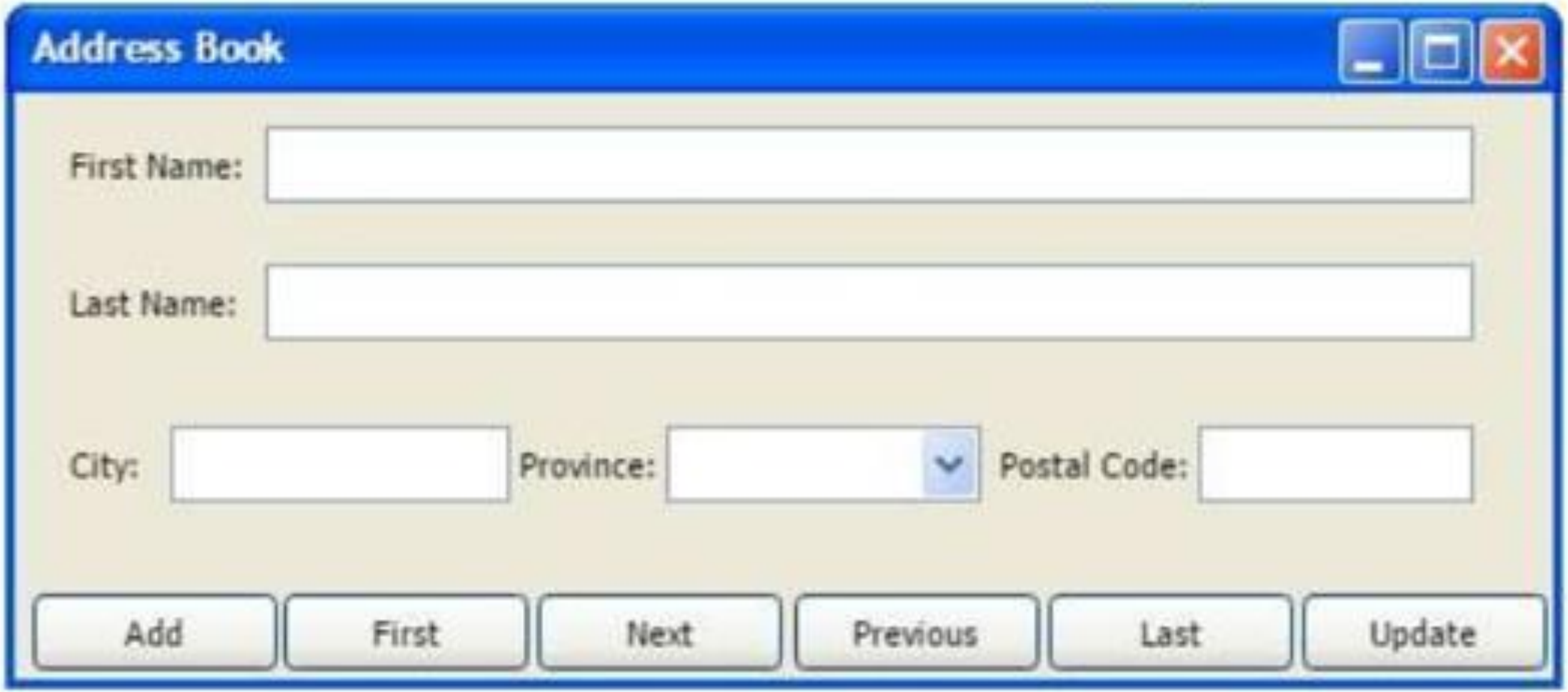
Mail Format:

HTML Plain Text Custom

OK Cancel



GUI Example-4



GUI Example-5