Lecture 7 - Java Graphical User Interface (GUI): JavaFX - Part IV

Guiliang Liu

The Chinese University of Hong Kong, Shenzhen

CSC-1004: Computational Laboratory Using Java
Course Page: [Click]

JavaFX Event Handling: Convenient Methods

- JavaFX provides convenient methods to **handle events** (create and register event handlers to respond to KeyEvent, MouseEvent, Action Event, and Drop Events).

- **Node** class contains various Event Handler properties which can be set to the user-defined Event Handlers using the **setter methods** defined in the class.

- Setting, EventHandler properties of the Node class, to the user defined event handlers, automatically **registers the handlers** to receive the event types.

JavaFX Event Handling: Convenient Methods

The EventHandler registered with the **setOnAction()** method is called when the Play button is clicked and it is set to rotate the rectangle on the screen.

```java
public void start(Stage primaryStage) {

    // Creating Rectangle
    Rectangle rect = new Rectangle(100,100,120,120);

    // Setting Stroke and colour for the rectangle
    rect.setFill(Color.RED);
    rect.setStroke(Color.BLACK);

    // Instantiating RotateTransition class
    RotateTransition rotate = new RotateTransition();

    //Setting properties for the Rotate Transition class
    rotate.setAutoReverse(false);
    rotate.setByAngle(360);
    rotate.setCycleCount(500);
    rotate.setDuration(Duration.millis(500));
    rotate.setNode(rect);

    //Creating the play button
    Button btn = new Button();
```

```java
    //Setting properties for the play button
    btn.setText("Play");
    btn.setTranslateX(100);
    btn.setTranslateY(250);

    //defining the convenience method to register the event
    btn.setOnAction(new EventHandler<ActionEvent>() {
        public void handle(ActionEvent event) {

            rotate.play();
        }
    });

    //Creating the pause button
    Button btn1 = new Button("Pause");

    //Setting propertied for the pause button
    btn1.setTranslateX(160);
    btn1.setTranslateY(250);
```

```java
    //Handling event for the pause button click event
    btn1.setOnAction(new EventHandler<ActionEvent>() {

        @Override
        public void handle(ActionEvent arg0) {
            // TODO Auto-generated method stub
            rotate.pause();
        }

    });

    //Configuring group and scene
    Group root = new Group();
    Scene scene = new Scene(root, 400, 350);
    root.getChildren().addAll(btn,rect,btn1);
    primaryStage.setScene(scene);
    primaryStage.setTitle("Handling Events");
    primaryStage.show();
}
```

g, Shenzhen

JavaFX Event Handling: Convenient Methods

The **setOnKeyEvent()** method can register the Event Handler logic for the key event.

E.g., the key pressed in the first text field is set as the text in the second text field.

```java
package application;
import javafx.application.Application;
import javafx.event.EventHandler;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.control.TextField;
import javafx.scene.input.KeyEvent;
import javafx.scene.paint.Color;
import javafx.stage.Stage;
public class JavaFX_KeyEvent extends Application{

    @Override
    public void start(Stage primaryStage) throws Exception {

        // TODO Auto-generated method stub

        //Creating TextFields and setting position for them
        TextField tf1 = new TextField();
        TextField tf2 = new TextField();
        tf1.setTranslateX(100);
        tf1.setTranslateY(100);
        tf2.setTranslateX(300);
        tf2.setTranslateY(100);
```
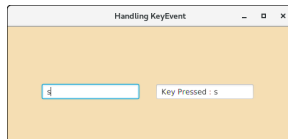
```java
//Handling KeyEvent for textfield 1
tf1.setOnKeyPressed(new EventHandler<KeyEvent>() {

    @Override
    public void handle(KeyEvent key) {
        // TODO Auto-generated method stub
        tf2.setText("Key Pressed :"+" "+key.getText());
    }

});

//setting group and scene
Group root = new Group();
root.getChildren().addAll(tf2,tf1);
Scene scene = new Scene(root,500,200,Color.WHEAT);
primaryStage.setScene(scene);
primaryStage.setTitle("Handling KeyEvent");
primaryStage.show();
}
public static void main(String[] args) {
    launch(args);
}
}

}
```


香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

JavaFX Event Handling: Event Handlers

- JavaFX facilitates us to use the **Event Handlers** to handle the events generated by Keyboard Actions, Mouse Actions, and many more source nodes.

- There can be more than one Event handlers for a single node.

- We can use single handler for more than one node and more than one event type.

香港中文大學 (深圳)
The Chinese University of Hong Kong, Shenzhen

JavaFX Event Handling: Event Handlers

In the following example, same event handler is registered with two different buttons.

The event source is discriminated in the handle() method.

```java
public void start(Stage primaryStage) throws Exception {
    // TODO Auto-generated method stub
    //Creating Circle and setting the color and stroke in the circle
    Circle c = new Circle(100,100,50);
    c.setFill(Color.GREEN);
    c.setStroke(Color.BLACK);

    //creating play button and setting coordinates for the button
    Button btn = new Button("Play");
    btn.setTranslateX(125);
    btn.setTranslateY(200);

    // creating pause button and setting coordinate for the pause button
    Button btn1 = new Button("Pause");
    btn1.setTranslateX(175);
    btn1.setTranslateY(200);

    //Instantiating TranslateTransition class to create the animation
    TranslateTransition trans = new TranslateTransition();

    //setting attributes for the TranslateTransition
    trans.setAutoReverse(true);
    trans.setByX(200);
    trans.setCycleCount(100);
    trans.setDuration(Duration.millis(500));
    trans.setNode(c);
```

```java
//Creating EventHandler
EventHandler<MouseEvent> handler = new EventHandler<MouseEvent>() {

    @Override
    public void handle(MouseEvent event) {
        // TODO Auto-generated method stub

        if(event.getSource()==btn)
        {
            trans.play(); //animation will be played when the play button is clicked
        }
        if(event.getSource()==btn1)
        {
            trans.pause(); //animation will be paused when the pause button is clicked
        }
        event.consume();
    }

};

//Adding Handler for the play and pause button
btn.setOnMouseClicked(handler);
btn1.setOnMouseClicked(handler);
```
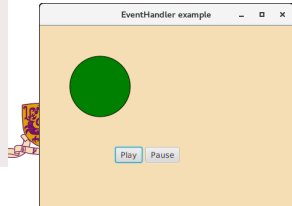
```java
//Creating Group and scene
Group root = new Group();
root.getChildren().addAll(c,btn,btn1);
Scene scene = new Scene(root,420,300,Color.WHEAT);
primaryStage.setScene(scene);
primaryStage.setTitle("EventHandler example");
primaryStage.show();
}
public static void main(String[] args) {
    launch(args);
}
}
}
```

Question and Answering (Q&A)