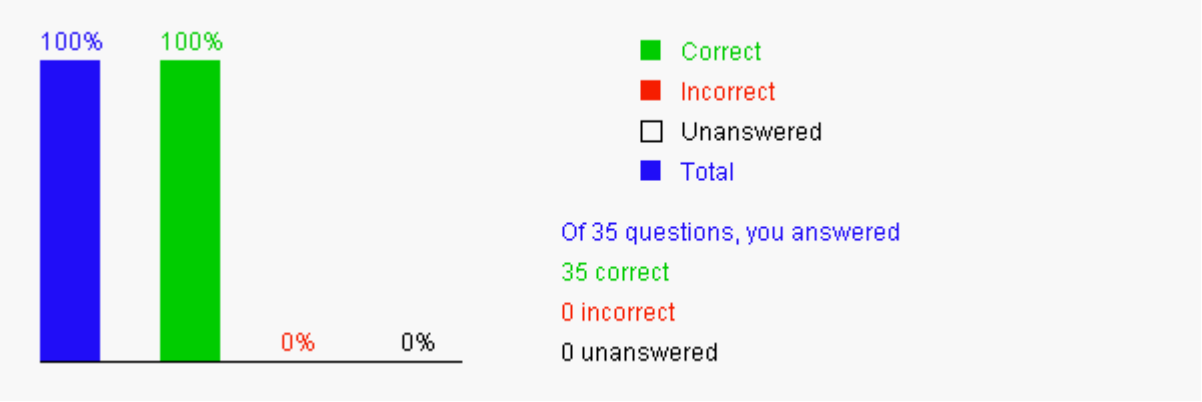
This quiz is for students to practice. A large number of additional quiz is available for instructors using Quiz Generator from the Instructor's Resource Website. Videos for Java, Python, and C++ can be found at https://yongdanielliang.github.io/revelvideos.html.

Chapter 15 Event-Driven Programming and Animations



Please send suggestions and errata to Dr. Liang at y.daniel.liang@gmail.com. Indicate which book and edition you are using. Thanks!

Section 15.2 Events and Event Sources

15.1 A JavaFX action event handler is an instance of _____.

- A. ActionEvent
- B. Action
- C. EventHandler
- D. EventHandler<ActionEvent> (selected)

Your answer is correct

15.2 A JavaFX action event handler contains a method _____.

- A. public void actionPerformed(ActionEvent e)
- B. public void actionPerformed(Event e)
- C. public void handle(ActionEvent e) (selected)
- D. public void handle(Event e)

Your answer is correct

15.3 A JavaFX event handler for event type T is an instance of _____.

- A. ActionEvent
- B. Action
- C. EventHandler
- D. EventHandler<T> (selected)

Your answer is correct

15.4 Which of the following are the classes in JavaFX for representing an event?

- [x] A. ActionEvent
- [x] B. MouseEvent
- [x] C. KeyEvent
- [x] D. WindowEvent

Your answer is correct

Section 15.3 Registering Handlers and Handling Events

15.5 To register a source for an action event with a handler, use _____.

- A. source.addAction(handler)
- B. source.setOnAction(handler) (selected)
- C. source.addOnAction(handler)
- D. source.setActionHandler(handler)

Your answer is correct

15.6 Which of the following statements are true?

- [ ] A. A handler object fires an event.

- [x] B. A source object fires an event.
- [ ] C. Any object such a String object can fire an event.
- [x] D. A handler is registered with the source object for processing the event.

Your answer is correct 

15.7 Which of the following statements are true?

- [x] A. A Button can fire an ActionEvent.
- [x] B. A Button can fire a MouseEvent.
- [x] C. A Button can fire a KeyEvent.
- [x] D. A TextField can fire an ActionEvent.

Your answer is correct 

15.8 Which of the following statements are true?

- [x] A. A Node can fire an ActionEvent.
- [x] B. A Node can fire a MouseEvent.
- [x] C. A Node can fire a KeyEvent.
- [x] D. A Scene can fire a MouseEvent.

Your answer is correct 

15.9 Which of the following statements are true?

- [ ] A. A Shape can fire an ActionEvent.
- [x] B. A Shape can fire a MouseEvent.
- [x] C. A Shape can fire a KeyEvent.
- [x] D. A Text is a Shape.
- [x] E. A Circle is a Shape.

Your answer is correct 

Section 15.4 Inner Classes

15.10 Which of the following statements are true?

- [x] A. Inner classes can make programs simple and concise.
- [x] B. An inner class can be declared public or private subject to the same visibility rules applied to a member of the class.
- [x] C. An inner class can be declared static. A static inner class can be accessed using the outer class name. A static inner class cannot access nonstatic members of the outer class.
- [x] D. An inner class supports the work of its containing outer class and is compiled into a class named OuterClassName$InnerClassName.class.

Your answer is correct 

15.11 Suppose A is an inner class in Test. A is compiled into a file named _____.

- ( ) A. A.class
- (•) B. Test$A.class
- ( ) C. A$Test.class
- ( ) D. Test&A.class

Your answer is correct 

15.12 Which statement is true about a non-static inner class?

- ( ) A. It must implement an interface.
- ( ) B. It is accessible from any other class.
- ( ) C. It can only be instantiated in the enclosing class.
- ( ) D. It must be final if it is declared in a method scope.
- (•) E. It can access private instance variables in the enclosing object.

Your answer is correct 

Section 15.5 Anonymous Class Handlers

15.13 Which of the following statements are true?

- [x] A. An anonymous inner class is an inner class without a name.
- [x] B. An anonymous inner class must always extend a superclass or implement an interface, but it cannot have an explicit extends or implements clause.

- [x] C. An anonymous inner class must implement all the abstract methods in the superclass or in the interface.
- [x] D. An anonymous inner class always uses the no-arg constructor from its superclass to create an instance. If an anonymous inner class implements an interface, the constructor is Object().
- [x] E. An anonymous inner class is compiled into a class named OuterClassName$n.class.

Your answer is correct 

15.14 Suppose A is an anonymous inner class in Test. A is compiled into a file named _____.

- ( ) A. A.class
- ( ) B. Test$A.class
- ( ) C. A$Test.class
- (•) D. Test$1.class
- ( ) E. Test&1.class

Your answer is correct 

15.15 Analyze the following code.

```
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.stage.Stage;

public class Test extends Application {
  @Override // Override the start method in the Application class
  public void start(Stage primaryStage) {
    Button btOK = new Button("OK");

    btOK.setOnAction(new EventHandler<ActionEvent>() {
      public void handle(ActionEvent e) {
        System.out.println("The OK button is clicked");
      }
    });

    Scene scene = new Scene(btOK, 200, 250);
    primaryStage.setTitle("MyJavaFX"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage
  }

  /**
   * The main method is only needed for the IDE with limited JavaFX
   * support. Not needed for running from the command line.
   */
  public static void main(String[] args) {
    launch(args);
  }
}
```

- ( ) A. The program has a compile error because no handlers are registered with btOK.
- ( ) B. The program has a runtime error because no handlers are registered with btOK.
- (•) C. The message "The OK button is clicked" is displayed when you click the OK button.
- ( ) D. The handle method is not executed when you click the OK button, because no handler is registered with btOK.

Your answer is correct 

15.16 Analyze the following code.

```
import javafx.application.Application;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.layout.HBox;
import javafx.stage.Stage;

public class Test extends Application {
  @Override // Override the start method in the Application class
  public void start(Stage primaryStage) {
    Button btOK = new Button("OK");
    Button btCancel = new Button("Cancel");

    EventHandler<ActionEvent> handler = new EventHandler<ActionEvent>() {
      public void handle(ActionEvent e) {
        System.out.println("The OK button is clicked");
      }
    };
```

```
    btOK.setOnAction(handler);
    btCancel.setOnAction(handler);

    HBox pane = new HBox(5);
    pane.getChildren().addAll(btOK, btCancel);

    Scene scene = new Scene(pane, 200, 250);
    primaryStage.setTitle("Test"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage
  }

  /**
   * The main method is only needed for the IDE with limited JavaFX
   * support. Not needed for running from the command line.
   */
  public static void main(String[] args) {
    launch(args);
  }
}
```

- [x] A. When clicking the OK button, the program displays The OK button is clicked.
- [x] B. When clicking the Cancel button, the program displays The OK button is clicked.
- [ ] C. When clicking either button, the program displays The OK button is clicked twice.
- [ ] D. The program has a runtime error, because the handler is registered with more than one source.

Your answer is correct 

Section 15.6 Simplifying Event Handing Using Lambda Expressions

15.17 Analyze the following code.

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

public class Test extends Application {
  @Override // Override the start method in the Application class
  public void start(Stage primaryStage) {
    // Create a button and place it in the scene
    Button btOK = new Button("OK");
    btOK.setOnAction(e -> System.out.println("OK 1"));
    btOK.setOnAction(e -> System.out.println("OK 2"));

    Scene scene = new Scene(btOK, 200, 250);
    primaryStage.setTitle("MyJavaFX"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage
  }

  /**
   * The main method is only needed for the IDE with limited JavaFX
   * support. Not needed for running from the command line.
   */
  public static void main(String[] args) {
    launch(args);
  }
}
```

- ( ) A. When clicking the button, the program displays OK1 OK2.
- ( ) B. When clicking the button, the program displays OK1.
- (•) C. When clicking the button, the program displays OK2.
- ( ) D. The program has a compile error, because the setOnAction method is invoked twice.

Your answer is correct 

15.18 Which of the following code correctly registers a handler with a button btOK?

- [x] A. btOK.setOnAction(e -> System.out.println("Handle the event"));
- [x] B. btOK.setOnAction((e) -> System.out.println("Handle the event"););
- [x] C. btOK.setOnAction((ActionEvent e) -> System.out.println("Handle the event"));
- [x] D. btOK.setOnAction(e -> {System.out.println("Handle the event");});

Your answer is correct 

15.19 Fill in the code below in the underline:

```
public class Test {
  public static void main(String[] args) {
```

```
    Test test = new Test();
    test.setAction(_____________________________);
  }

  public void setAction(T1 t) {
    t.m();
  }
}

interface T1 {
  public void m();
}
```

- (●) A. () -> System.out.print("Action 1! ")
- ( ) B. (e) -> System.out.print("Action 1! ")
- ( ) C. System.out.print("Action 1! ")
- ( ) D. (e) -> {System.out.print("Action 1! ")}

Your answer is correct 

15.20 Fill in the code below in the underline:

```
public class Test {
  public static void main(String[] args) {
    Test test = new Test();
    test.setAction2(_____________________________);
  }

  public void setAction2(T2 t) {
    t.m(4.5);
  }
}

interface T2 {
  public void m(Double d);
}
```

- [ ] A. () -> System.out.print(e)
- [x] B. (e) -> System.out.print(e)
- [x] C. e -> System.out.print(e)
- [x] D. (e) -> {System.out.print(e);}

Your answer is correct 

15.21 Fill in the code below in the underline:

```
public class Test {
  public static void main(String[] args) {
    Test test = new Test();
    System.out.println(test.setAction3(_____________));
  }

  public double setAction3(T3 t) {
    return t.m(5.5);
  }
}

interface T3 {
  public double m(Double d);
}
```

- [ ] A. () -> e * 2
- [x] B. (e) -> e * 2
- [x] C. e -> e * 2
- [ ] D. (e) -> {e * 2;}

Your answer is correct 

15.22 Analyze the following code:

```
public class Test {
  public static void main(String[] args) {
    Test test = new Test();
    test.setAction(() -> System.out.print("Action 1! "));
  }

  public void setAction(T t) {
    t.m1();
  }
}
```

```
interface T {
  public void m1();
  public void m2();
}
```

- [ ] A. The program displays Action 1.
- [x] B. The program has a compile error because T is not a functional interface. T contains multiple methods.
- [x] C. The program would work if you delete the method m2 from the interface T.
- [ ] D. The program has a runtime error because T is not a functional interface. T contains multiple methods.

Your answer is correct

Section 15.8 Mouse Events

15.23 To handle the mouse click event on a pane p, register the handler with p using _____.

- (•) A. p.setOnMouseClicked(handler);
- ( ) B. p.setOnMouseDragged(handler);
- ( ) C. p.setOnMouseReleased(handler);
- ( ) D. p.setOnMousePressed(handler);

Your answer is correct

15.24 Fill in the code in the underlined location to display the mouse point location when the mouse is pressed in the pane.

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

public class Test extends Application {
  @Override // Override the start method in the Application class
  public void start(Stage primaryStage) {
    Pane pane = new Pane();
    ____________________________________

    Scene scene = new Scene(pane, 200, 250);
    primaryStage.setTitle("Test"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage
  }

  /**
   * The main method is only needed for the IDE with limited JavaFX
   * support. Not needed for running from the command line.
   */
  public static void main(String[] args) {
    launch(args);
  }
}
```

- ( ) A. pane.setOnMouseClicked((e) -> System.out.println(e.getX() + ", " + e.getY()));
- ( ) B. pane.setOnMouseReleased(e -> {System.out.println(e.getX() + ", " + e.getY())});
- (•) C. pane.setOnMousePressed(e -> System.out.println(e.getX() + ", " + e.getY()));
- ( ) D. pane.setOnMouseDragged((e) -> System.out.println(e.getX() + ", " + e.getY()));

Your answer is correct

Section 15.9 Key Events

15.25 To handle the key pressed event on a pane p, register the handler with p using _____.

- ( ) A. p.setOnKeyClicked(handler);
- ( ) B. p.setOnKeyTyped(handler);
- ( ) C. p.setOnKeyReleased(handler);
- (•) D. p.setOnKeyPressed(handler);

Your answer is correct

15.26 Fill in the code to display the key pressed in the text.

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.text.Text;
import javafx.stage.Stage;

public class Test extends Application {
  @Override // Override the start method in the Application class
```

```
  public void start(Stage primaryStage) {
    Pane pane = new Pane();
    Text text = new Text(20, 20, "Welcome");
    pane.getChildren().add(text);
    Scene scene = new Scene(pane, 200, 250);
    primaryStage.setTitle("Test"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage

    text.setFocusTraversable(true);
    text.setOnKeyPressed(_____________________);
  }

  /**
   * The main method is only needed for the IDE with limited JavaFX
   * support. Not needed for running from the command line.
   */
  public static void main(String[] args) {
    launch(args);
  }
}
```

- [ ] A. () -> text.setText(e.getText())
- [x] B. (e) -> text.setText(e.getText())
- [x] C. e -> text.setText(e.getText())
- [x] D. e -> {text.setText(e.getText());}

Your answer is correct 

15.27 Suppose the following program displays a pane in the stage. What is the output if the user presses the key for letter B?

```
import javafx.application.Application;
import static javafx.application.Application.launch;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.stage.Stage;

// import javafx classes omitted
public class Test1 extends Application {
  @Override
  public void start(Stage primaryStage) {
    // Code to create and display pane omitted
    Pane pane = new Pane();
    Scene scene = new Scene(pane, 200, 250);
    primaryStage.setTitle("MyJavaFX"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage

    pane.requestFocus();
    pane.setOnKeyPressed(e ->
      System.out.print("Key pressed " + e.getCode() + " "));
    pane.setOnKeyTyped(e ->
      System.out.println("Key typed " + e.getCode()));
  }

  /**
   * The main method is only needed for the IDE with limited
   * JavaFX support. Not needed for running from the command line.
   */
  public static void main(String[] args) {
    launch(args);
  }
}
```

- (●) A. Key pressed B Key typed UNDEFINED
- ( ) B. Key pressed B Key typed
- ( ) C. Key typed UNDEFINED
- ( ) D. Key pressed B

Your answer is correct 

15.28 Suppose the follwoing program displays a pane in the stage. What is the output if the user presses the DOWN arrow key?

```
import javafx.application.Application;
import static javafx.application.Application.launch;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.stage.Stage;

// import javafx classes omitted
public class Test1 extends Application {
```

```
  @Override
  public void start(Stage primaryStage) {
    // Code to create and display pane omitted
    Pane pane = new Pane();
    Scene scene = new Scene(pane, 200, 250);
    primaryStage.setTitle("MyJavaFX"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage

    pane.requestFocus();
    pane.setOnKeyPressed(e ->
      System.out.print("Key pressed " + e.getCode() + " "));
    pane.setOnKeyTyped(e ->
      System.out.println("Key typed " + e.getCode()));
  }

  /**
   * The main method is only needed for the IDE with limited
   * JavaFX support. Not needed for running from the command line.
   */
  public static void main(String[] args) {
    launch(args);
  }
}
```

- ○ A. Key pressed DOWN Key typed UNDEFINED
- ○ B. Key pressed DOWN Key typed
- ○ C. Key typed UNDEFINED
- ◉ D. Key pressed DOWN

Your answer is correct 

Section 15.10 Listeners for Observable Objects

15.29 Analyze the following code:

```
import javafx.beans.property.DoubleProperty;
import javafx.beans.property.SimpleDoubleProperty;

public class Test {
  public static void main(String[] args) {
    DoubleProperty balance = new SimpleDoubleProperty();
    balance.addListener(ov ->
      System.out.println(2 + balance.doubleValue()));

    balance.set(4.5);
  }
}
```

- ○ A. The program displays 4.5.
- ◉ B. The program displays 6.5.
- ○ C. The program would display 4.5 if the balance.set(4.5) is placed before the balance.addListener(...) statement.
- ○ D. The program would display 6.5 if the balance.set(4.5) is placed before the balance.addListener(...) statement.

Your answer is correct 

Section 15.11 Animation

15.30 Which of the following methods is not defined in the Animation class?

- ○ A. pause()
- ○ B. play()
- ○ C. stop()
- ◉ D. resume()

Your answer is correct 

15.31 The properties _____ are defined in the Animation class.

- ☑ A. autoReverse
- ☑ B. cycleCount
- ☑ C. rate
- ☑ D. status

Your answer is correct 

15.32 The properties _____ are defined in the PathTransition class.

- ☑ A. duration
- ☑ B. node

- [x] C. orientation
- [x] D. path

Your answer is correct 

15.33 To properties _____ are defined in the FadeTransition class.

- [x] A. duration
- [x] B. node
- [x] C. fromValue
- [x] D. toValue
- [x] E. byValue

Your answer is correct 

15.34 To create a KeyFrame with duration 1 second, use _____.

- [ ] A. new KeyFrame(1000, handler)
- [ ] B. new KeyFrame(1, handler)
- [x] C. new KeyFrame(Duration.millis(1000), handler)
- [x] D. new KeyFrame(Duration.seconds(1), handler)

Your answer is correct 

15.35 _____ is a subclass of Animation.

- [x] A. PathTransition
- [x] B. FadeTransition
- [x] C. Timeline
- [ ] D. Duration

Your answer is correct