//////////////////////////// 1. SETTING DRAGGED IN MouseEventDemo.java

pane.getChildren().addAll(text);

text.setOnMouseDragged( e -> {

text.setX(e.getX());

text.setY(e.getY());

});

///////////////////////////// 2. SETTING DRAGGED WITH ARROW KEYS in KeyEventDemo.java

text.setOnKeyPressed( e -> {

switch(e.getCode()){

case *UP*:

text.setY(text.getY()-10);

break;

case *DOWN*:

text.setY(text.getY()+10);

break;

case *LEFT*:

text.setX(text.getX()-10);

break;

case *RIGHT*:

text.setX(text.getX()+10);

break;

default:

if (e.getText().length()>0)

text.setText(e.getText());

}

});

Scene scene = new Scene(pane, 300, 300);

keyStage.setTitle("KeyEvent Demo");

keyStage.setScene(scene);

keyStage.show();

// object must have focus to recognize events

///////////////////////////////////// 3. We have to focus to active to response to the event

text.requestFocus();

package COSC60220220211;

import javafx.beans.InvalidationListener;

import javafx.beans.Observable;

import javafx.beans.property.DoubleProperty;

import javafx.beans.property.IntegerProperty;

import javafx.beans.property.SimpleDoubleProperty;

public class C3ObservablePropertyDemo {

public static void main(String[] args) {

DoubleProperty balance= new SimpleDoubleProperty();

balance.addListener(new InvalidationListener() {

@Override

public void invalidated(Observable observable) {

System.*out*.println("The new value of balance is " + balance.doubleValue());

}

});

balance.set(4.5);

System.*out*.println();

System.*out*.println("Some thing else here");

System.*out*.println();

balance.set(10.5);

}

}

package COSC60220220211;

//////////////////////////////////Listing 15.12 ResizableCircleRectangle.java

import javafx.application.Application;

import javafx.scene.paint.Color;

import javafx.scene.shape.Circle;

import javafx.scene.shape.Rectangle;

import javafx.stage.Stage;

import javafx.scene.Scene;

import javafx.scene.control.Label;

import javafx.scene.layout.StackPane;

public class C4ResizeableCircleWindow extends Application {

private Circle circle = new Circle(60);

private Rectangle rectangle = new Rectangle(120, 120);

// Place clock and label in border pane

private StackPane pane = new StackPane();

@Override // Override the start method in the Application class

public void start(Stage stage) {

circle.setFill(Color.*DARKTURQUOISE*);

rectangle.setFill(Color.*WHITE*);

rectangle.setStroke(Color.*BLACK*);

pane.getChildren().addAll(rectangle, circle);

// Create a scene and place the pane in the stage

Scene scene = new Scene(pane, 140, 140);

stage.setTitle("Resizable Window");

stage.setScene(scene); // Place the scene in the stage

stage.show(); // Display the stage

pane.widthProperty().addListener(ov -> resize()); //Lambda Version

/\*

same as C3Obse.....

Line 15-20

new InvalidationListener() {

@Override

public void invalidated(Observable observable) {

System.out.println("The new value of balance is " + balance.doubleValue());

}

}

is EQUAL ov -> resize()

\*/

pane.heightProperty().addListener(ov -> resize());

}

private void resize() {

double length = Math.*min*(pane.getWidth(), pane.getHeight());

circle.setRadius(length / 2 - 15);

rectangle.setWidth(length - 30);

rectangle.setHeight(length -30);

}

}

CHAPTER 16

