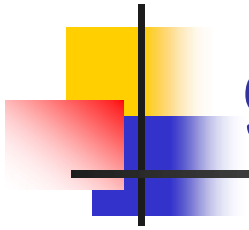
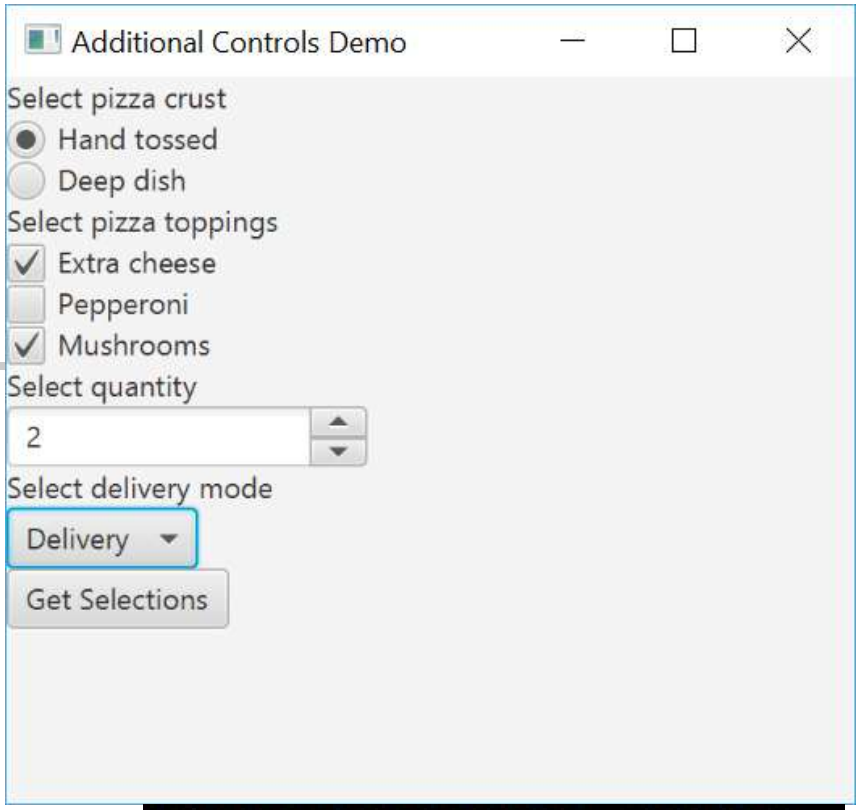


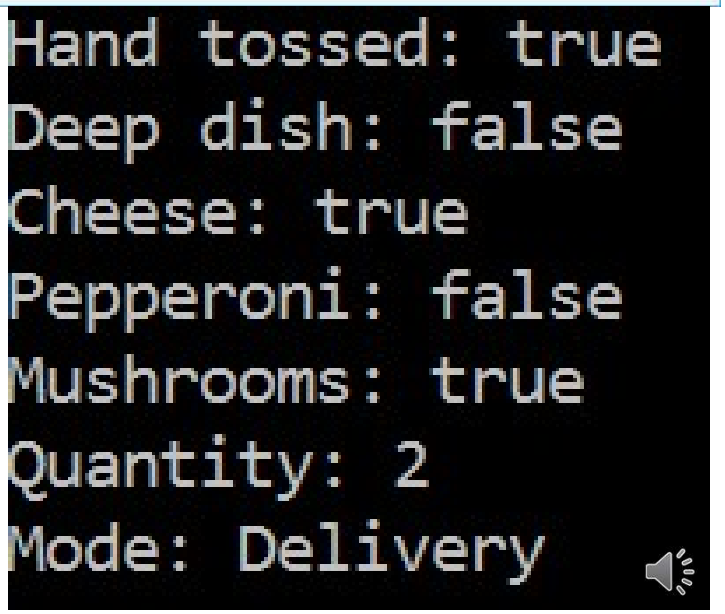
9.4 Graphics Supplement: Outline

- Additional User Interface Controls and Shapes
- Handling Mouse Events
- Introduction to the Timeline

Additional User Controls

- CheckBox, RadioButton (in ToggleGroup), Spinner, and ChoiceBox demo
- Example) selecting pizza options
- listing 9.13 **class AdditionalControlsDemo**

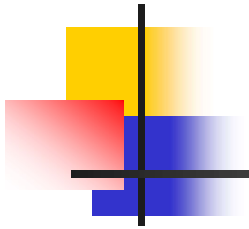




listing 9.13, **AdditionalControlsDemo**

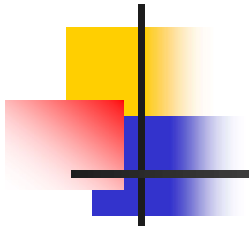
```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.stage.Stage;
import javafx.scene.layout.VBox;
import javafx.scene.control.Button;
import javafx.scene.control.CheckBox;
import javafx.scene.control.RadioButton;
import javafx.scene.control.ToggleGroup;
import javafx.scene.control.Spinner;
import javafx.scene.control.SpinnerValueFactory;
import javafx.scene.control.ChoiceBox;
import javafx.collections.FXCollections;
import javafx.scene.control.Label;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;

/**
Simple demonstration of some additional JavaFX
UI controls.
*/
public class AdditionalControlsDemo extends Application
{
   public static void main(String[] args)
   {
      launch(args);
   }
```

```
@Override
  public void start(Stage primaryStage) throws Exception
  {
        VBox root = new VBox();

        // Demonstrate radio buttons
        root.getChildren().add(new Label("Select pizza crust"));
        ToggleGroup toggleCrust = new ToggleGroup();
        RadioButton rbHand = new RadioButton("Hand tossed");
        rbHand.setToggleGroup(toggleCrust);
        rbHand.setSelected(true);

        RadioButton rbDeepDish = new RadioButton("Deep dish");
        rbDeepDish.setToggleGroup(toggleCrust);
        root.getChildren().add(rbHand);
        root.getChildren().add(rbDeepDish);

        // Demonstrate checkboxes
        root.getChildren().add(new Label("Select pizza toppings"));
        CheckBox cbCheese = new CheckBox("Extra cheese");
        CheckBox cbPepperoni = new CheckBox("Pepperoni");
        CheckBox cbMushrooms = new CheckBox("Mushrooms");
        root.getChildren().add(cbCheese);
        root.getChildren().add(cbPepperoni);
        root.getChildren().add(cbMushrooms);
```



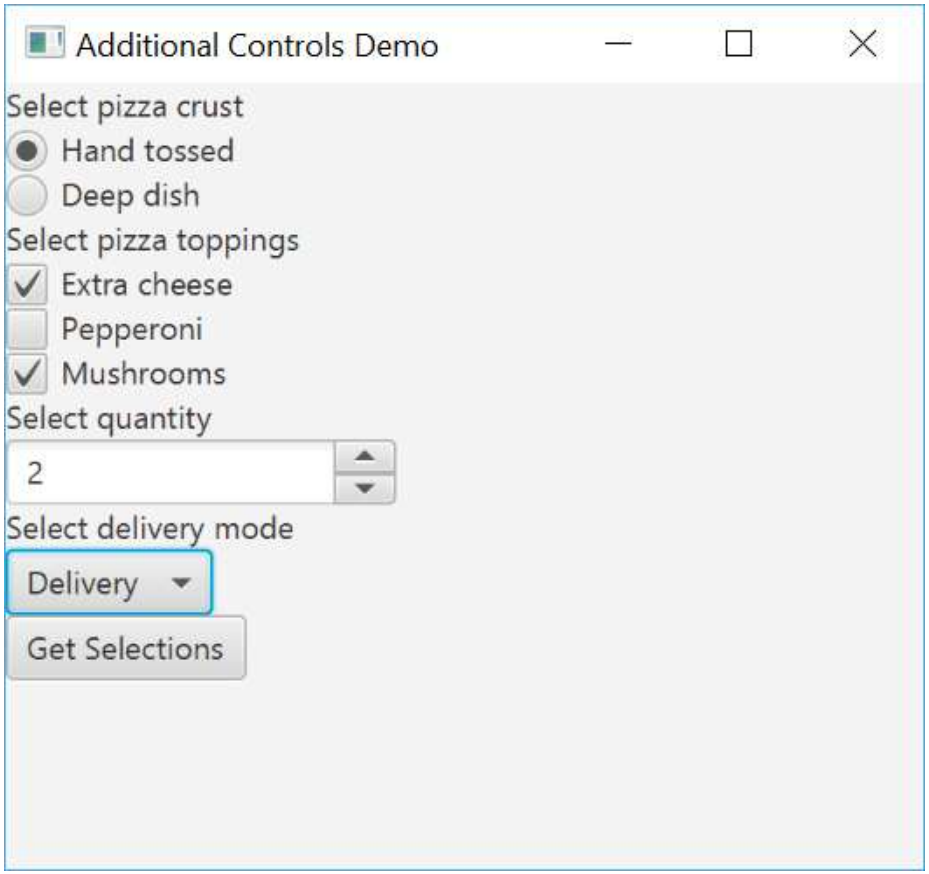
- ToggleGroup
  - 라디오 버튼이 체크 박스와 다른 것은 여러 컨트롤이 함께 다루어진다는 점에 있음.
  - 라디오 버튼은 여러 버튼이 그룹으로 기능하고 그 중에서 항상 하나만 선택되도록 해야 함
  - 그래서 "ToggleGroup"라는 클래스를 이용
  - ON/OFF하는 여러 컨트롤을 하나의 그룹으로 관리하는 기능을 제공

```
        // Demonstrate Spinner with integer values from 1-10
        root.getChildren().add(new Label("Select quantity"));
        Spinner<Integer> spinnerQuantity = new Spinner();
        final int defaultValue = 1;
        // Value factory.
        SpinnerValueFactory<Integer> quantityFactory =
      new SpinnerValueFactory.IntegerSpinnerValueFactory
            (1, 10, defaultValue);
        spinnerQuantity.setValueFactory(quantityFactory);

        root.getChildren().add(spinnerQuantity);

        // Demonstrate ChoiceBox with delivery options
        root.getChildren().add(new Label("Select delivery mode"));

        ChoiceBox<String> cbModes = new ChoiceBox<String>(
          FXCollections.observableArrayList("Delivery",
                        "Dine-In", "Carryout"));

root.getChildren().add(cbModes);

// Button to display selections
        Button btnSelections = new Button("Get Selections");
```



- Spinner
  - Spinner는 ComboBox와 비슷하지만 드롭다운이 없이 현재 데이터 값을 나타내며 증가, 감소 버튼으로 값을 변경할 수 있는 컨트롤
  - 순차적인 데이터(수치등)를 나타낼 때 주로 사용
  - 정렬되어 있는 값(숫자, 객체)을 선택할 수 있는 단일 행 텍스트 필드
  - 숫자, 객체 값을 단계적으로 선택할 수 있는 화살표(Up, Down) 버튼을 제공

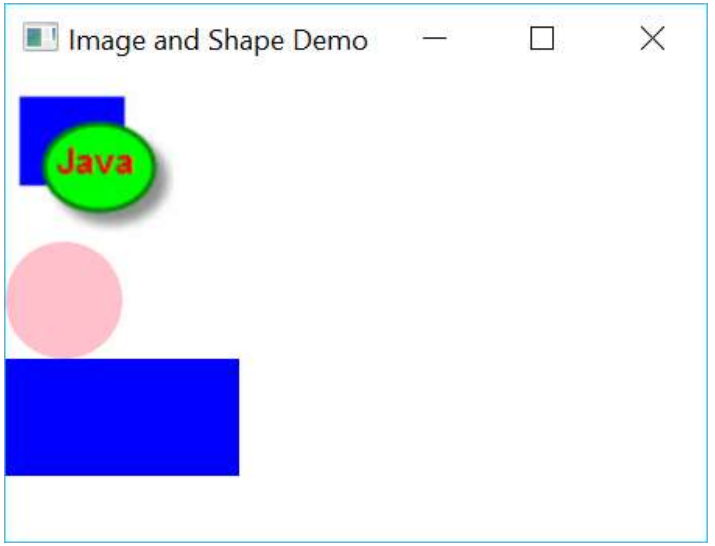
```
            // Set the event handler when the button is clicked
            btnSelections.setOnAction(new EventHandler<ActionEvent>()
            {
                        @Override
                        public void handle(ActionEvent event)
                        {
                                    System.out.println("Hand tossed: " +
rbHand.isSelected());
                                    System.out.println("Deep dish: " +
                        rbDeepDish.isSelected());
                                    System.out.println("Cheese: " + cbCheese.isSelected());
                                    System.out.println("Pepperoni: " +
                        cbPepperoni.isSelected());
                                    System.out.println("Mushrooms: " +
                        cbMushrooms.isSelected());
                                    System.out.println("Quantity: " +
                        spinnerQuantity.getValue());
                                    System.out.println("Mode: " + cbModes.getValue());
                        }
              }
            );
            root.getChildren().add(btnSelections);

            Scene scene = new Scene(root, 350, 300);
            primaryStage.setTitle("Additional Controls Demo");
            primaryStage.setScene(scene);
            primaryStage.show();
   }
}
```




```
Hand tossed: true
Deep dish: false
Cheese: true
Pepperoni: false
Mushrooms: true
Quantity: 2
Mode: Delivery
```

Images and Shapes

- We can load and display images with the **`Image`** object placed into an **`ImageView`** and added to the pane
- Shape objects can be manipulated unlike drawing shapes using stroke/fill
- listing 9.14 **`class ImageShapeDemo`**



- listing 9.14, **ImageShapeDemo**

```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.stage.Stage;
import javafx.scene.layout.VBox;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.scene.shape.Circle;
import javafx.scene.shape.Rectangle;
import javafx.scene.paint.Color;

/**
Demonstration of some shapes and an image
within a VBox layout.
*/
public class ImageShapeDemo extends Application
{
   public static void main(String[] args)
   {
      launch(args);
   }
```

```
@Override
  public void start(Stage primaryStage) throws Exception
  {
            VBox root = new VBox();
            ImageView imv = new ImageView();
            // Java looks for "java.jpg" in the default folder
            Image img = new Image("java.jpg");
            imv.setImage(img);

            Circle c = new Circle();
            c.setRadius(25);
            c.setFill(Color.PINK);

            Rectangle r = new Rectangle();
            r.setWidth(100);
            r.setHeight(50);
            r.setFill(Color.BLUE);

            root.getChildren().add(imv);
            root.getChildren().add(c);
            root.getChildren().add(r);


            Scene scene = new Scene(root, 300, 200);
            primaryStage.setTitle("Image and Shape Demo");
            primaryStage.setScene(scene);
            primaryStage.show();
  }
}
```

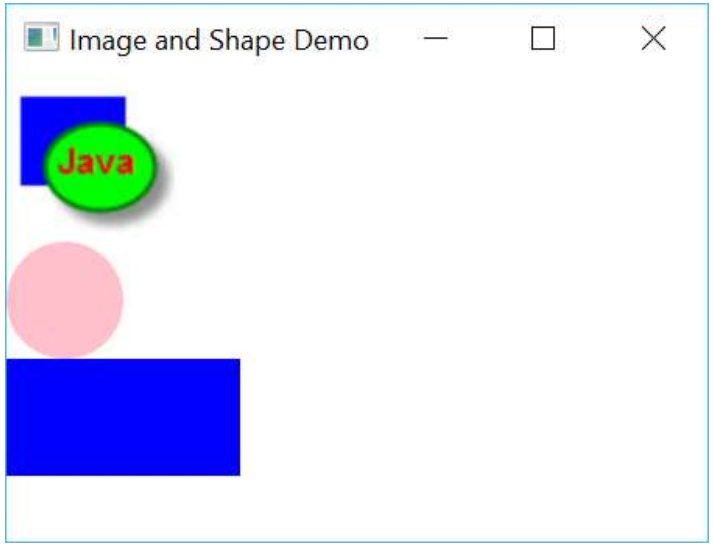


Image and Shape Demo
Java

Mouse Click and Mouse Move Events

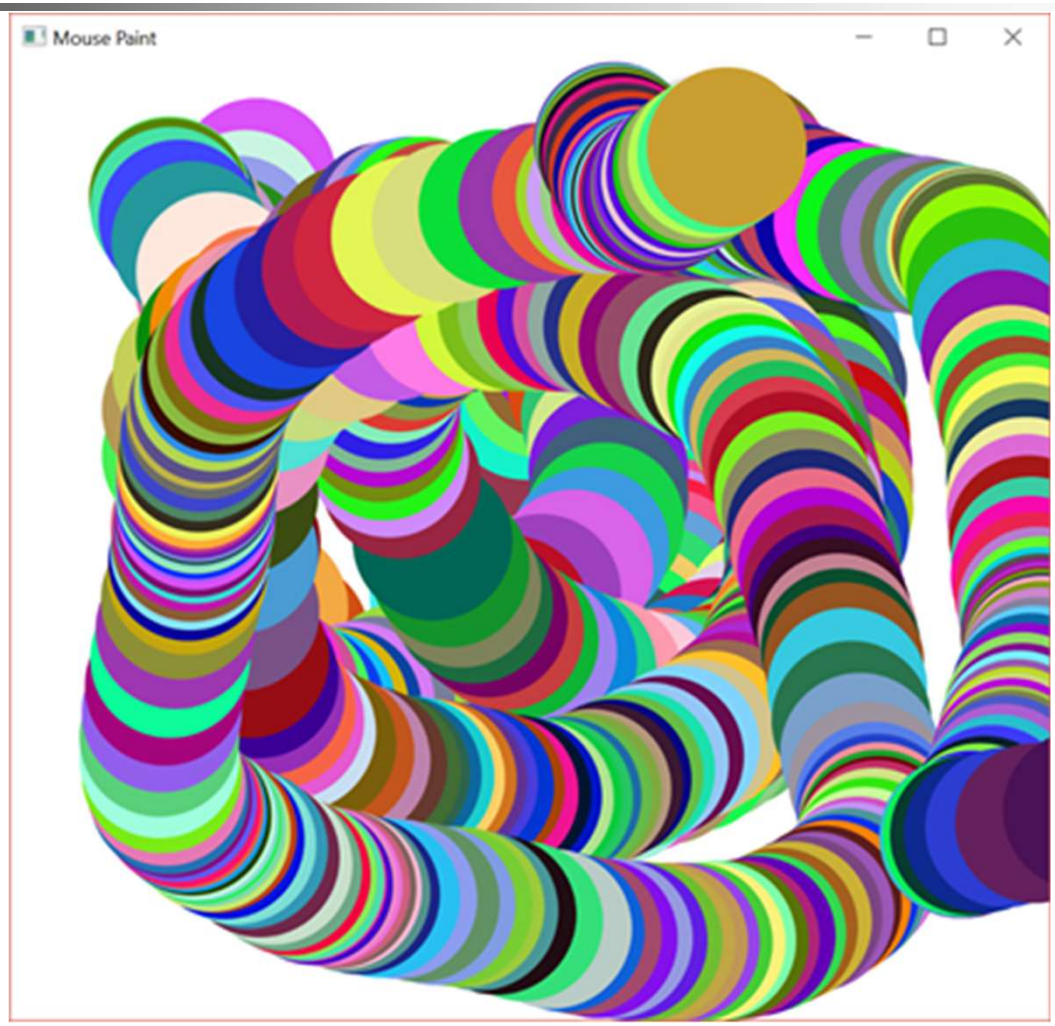
- Mouse events are handled in essentially the same manner that we used to handle button clicks except we implement **`EventHandler<MouseEvent>`** and override **`setOnMousePressed`** for mouse clicks and **`setOnMousePressed`** for mouse motion
- listing 9.15, **`class MousePaint`**

- listing 9.15, **MousePaint**

```
import javafx.application.Application;
import javafx.scene.canvas.Canvas; //
import javafx.scene.Scene;
import javafx.stage.Stage;
import javafx.scene.Group;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;
import javafx.scene.input.MouseEvent;
import javafx.event.EventHandler;
import java.util.Random;

/**
This program draws a circle of random color
at the location of the mouse whenever the mouse
moves.
*/
public class MousePaint extends Application
{
   private Random rnd = new Random();

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

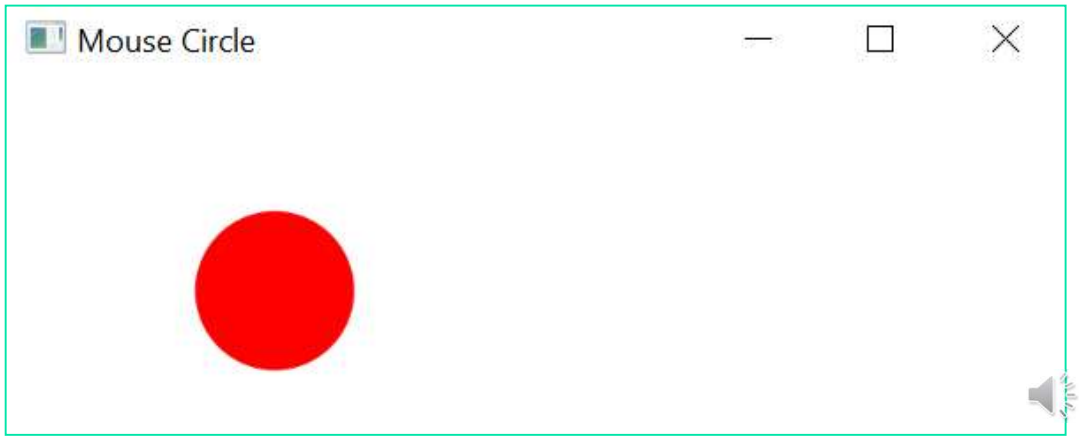
```
@Override
 public void start(Stage primaryStage) throws Exception
 {
        Group root = new Group();
        Canvas canvas = new Canvas(650, 600);
        GraphicsContext gc = canvas.getGraphicsContext2D();

        // When the mouse is pressed erase the
        // screen by drawing a white rectangle on the
        // entire canvas
        canvas.setOnMousePressed(new EventHandler<MouseEvent>()
        {
    @Override
    public void handle(MouseEvent event)
    {
        gc.setFill(Color.WHITE);
        gc.fillRect(0,0,canvas.getWidth(),canvas.getHeight());
    }
        });
```

```
            // When the mouse is moved get a random color
            // and draw a circle at the mouse's coordinates
            canvas.setOnMouseMoved(new EventHandler<MouseEvent>()
            {
        @Override
                        public void handle(MouseEvent event)
        {
                // Get a random color
          gc.setFill(Color.rgb(rnd.nextInt(255),
              rnd.nextInt(255),
              rnd.nextInt(255)));
          gc.fillOval(event.getX(),event.getY(),100,100);
        }
            });

            root.getChildren().add(canvas);
            primaryStage.setScene(new Scene(root));
            primaryStage.setTitle("Mouse Paint");
            primaryStage.show();
    }
}
```


Mouse Paint

Moving a Circle Object to th e Mouse Location

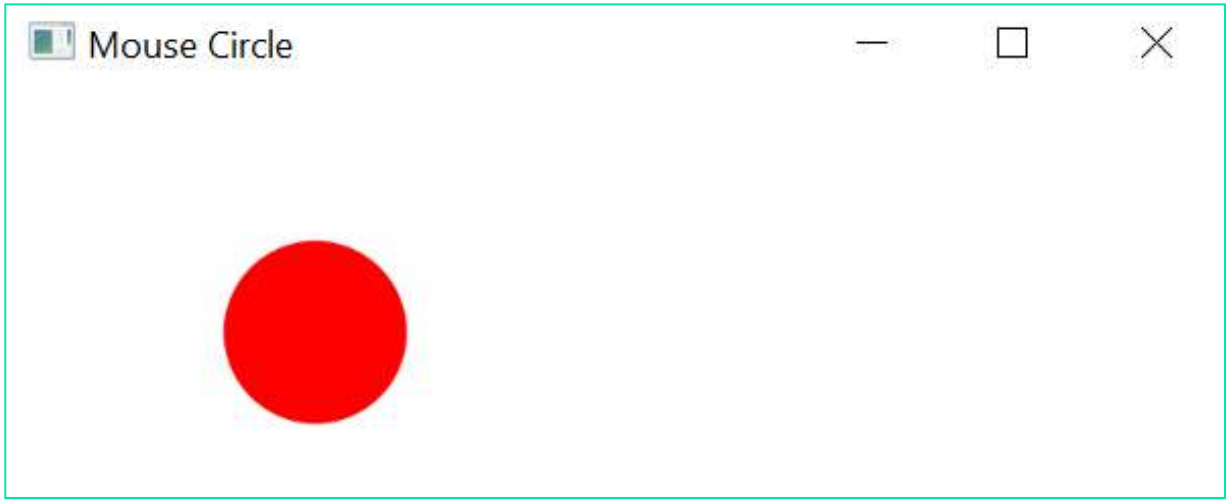
- Instead of painting circles this program moves a Circle object to the coordinate s of the mouse every time the mouse is moved
- View mouse demo2, listing 9.16, **`class MouseCircle`**



- listing 9.16, **MouseCircle**

```
import javafx.application.Application;
import javafx.scene.canvas.Canvas;
import javafx.scene.Scene;
import javafx.stage.Stage;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.input.MouseEvent;
import javafx.event.EventHandler;
import javafx.scene.shape.Circle;

/**
This program sets the X/Y coordinates of a Circle to the
location of the mouse.
*/
public class MouseCircle extends Application
{
   public static void main(String[] args)
   {
      Application.launch(args);
   }
```

```
@Override
  public void start(Stage primaryStage) throws Exception
  {
            Pane root = new Pane();
            root.setPrefSize(400,400);

            Circle circle = new Circle();
            circle.setRadius(30);
            circle.setFill(Color.RED);

            root.setOnMouseMoved(new EventHandler<MouseEvent>()
            {
            @Override
            public void handle(MouseEvent event)
            {
                        circle.setCenterX(event.getX());
                        circle.setCenterY(event.getY());
            }
            });

            root.getChildren().add(circle);
            primaryStage.setScene(new Scene(root));
            primaryStage.setTitle("Mouse Circle");
            primaryStage.show();
  }
}
```


Mouse Circle

끝