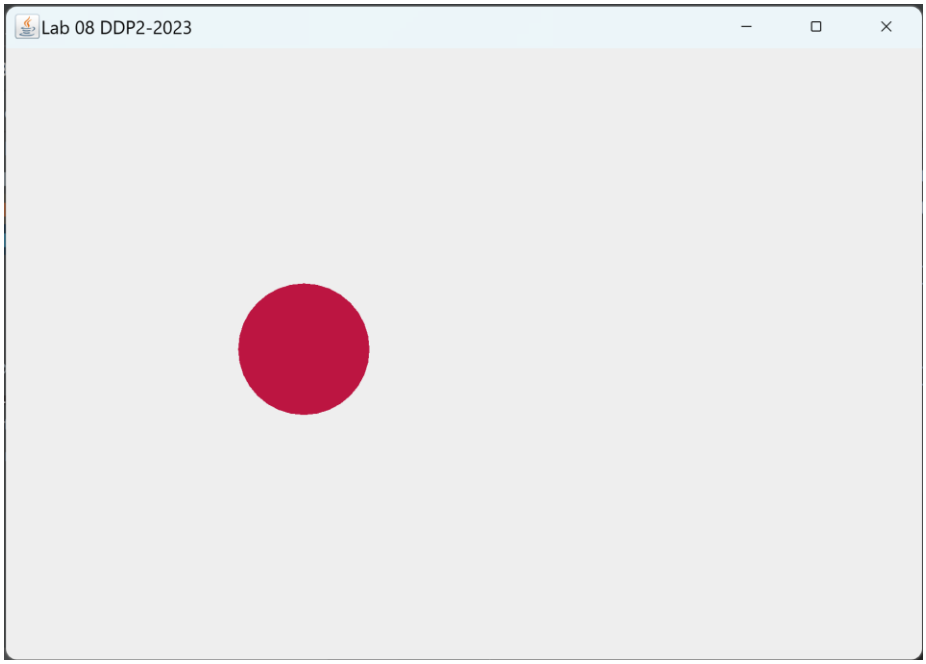
CSGE601021 Foundations of Programming 2
Lab 08: Working with Mouse Events

Build a Java project for this lab. Export this project as a *runnable JAR file* which can be executed outside your IDE.

Task Description:

Drawing, Moving, Dragging & Deleting Circles with Mouse Clicks



File *Lab08.java* sets up a panel that creates and draws a circle defined in *Circle.java* of random size and color at each mouse click. Each circle replaces the one before it. The code to handle the mouse clicks and do the drawing is in *CirclePanel.java*. Compile them and run them and experiment with the GUI. Then modify these files as described below.

1. This program creates a new circle each time—you can tell because each circle is a different color and size. Write a method ***void move(Point p)*** for your **Circle** class that takes a Point and moves the circle so its center is at that point. Now modify your CirclesListener class (defined inside CirclePanel) so that instead of creating a new circle every time the user clicks, it moves the existing circle to the clickpoint if a circle already exists. If no circle exists, a new one should be created at the clickpoint. So now a circle of the same color and size should move around the screen.

2. Write a method ***boolean isInside(Point p)*** for your **Circle** class that takes a Point and tells whether it is inside the circle. A point is inside the circle if its distance from the center is less than the radius. (Recall that the Euclidean distance between two points (x1,y1) and (x2,y2) is $\text{sqrt}((x2-x1)^2+(y2-y1)^2$.)

3. Now modify the ***mouseClicked*** method of CirclesListener so that the GUI behaves as follows:
   - If there is no circle (i.e., it is **null**) and the user clicks anywhere, a new (random) circle should be drawn at the click point.

- If there is a circle on the screen and the user clicks inside that circle, the circle should go away. (Hint: To make the circle go away, set it to **null** and **repaint**.)
- If there is a circle on the screen and the user clicks somewhere else, the circle should move to that point (no change from before).

So the logic for ***mouseClicked*** should look like this:

```
if there is currently no circle
   create a new circle at the click point
else if the click is inside a circle
   make the circle go away (i.e. delete the circle)
else
   move the circle to the click point

repaint
```

4. Add bodies for the ***mouseEntered*** and ***mouseExited*** methods so that when the mouse enters the panel the background turns **white**, and when it exits the background turns **black**. Remember that you can set the background color with the ***setBackground*** method.

5. Modify the code in CirclePanel.java so that the user can **drag** an *existing circle* around as long as the mouse button is depressed (pushed down). You will need to make the following changes:

   a) In the CirclePanel constructor, create a CirclesListener object and make it listen for both mouse events and mouse motion events.

   b) Make the CirclesListener class implement the **MouseMotionListener** interface in addition to the **MouseListener** interface. Add bodies for the two MouseMotionListener methods, *mouseDragged* and *mouseMoved*. In *mouseDragged*, simply move the existing circle (if any) to the point returned by the *getPoint* method of the MouseEvent and repaint. Provide an empty body for *mouseMoved*.

```
//********************************************************************
//  Lab08.java
//
//  Demonstrates mouse events and drawing on a panel.
//********************************************************************

package lab08_2023;

import javax.swing.JFrame;

public class Lab08 {
   //-----------------------------------------------------------------
   //  Creates and displays the application frame.
   //-----------------------------------------------------------------
   public static void main (String[] args) {
      JFrame circlesFrame = new JFrame ("Lab 08 DDP2-2023");
      circlesFrame.setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);

      circlesFrame.getContentPane().add (new CirclePanel());

      circlesFrame.pack();
      circlesFrame.setVisible(true);
   }
}
```

```
// ************************************************************
// Circle.java
//
// Define a Circle class with methods to create and draw
// a circle of random size, color, and location.
// ************************************************************

package lab08_2023;

import java.awt.*;
import java.util.Random;

public class Circle {
    private int centerX, centerY;
    private int radius;
    private Color color;

    static Random generator = new Random();

    //-----------------------------------------------------------
    // Creates a circle with center at point given, random radius and color
    //   -- radius 25..74
    //   -- color RGB value 0..16777215 (24-bit)
    //-----------------------------------------------------------
    public Circle(Point point) {
      radius = Math.abs(generator.nextInt())%50 + 25;
      color = new Color(Math.abs(generator.nextInt())% 16777216);
      centerX = point.x;
      centerY = point.y;
    }

    //-----------------------------------------------------------
    // Draws circle on the graphics object given
    //-----------------------------------------------------------
    public void draw(Graphics page) {
      page.setColor(color);
      page.fillOval(centerX-radius,centerY-radius,radius*2,radius*2);
    }
}
```

```
//********************************************************************
//  CirclePanel.java
//  Represents the primary panel for the Lab08 program on which the
//  circles are drawn.
//********************************************************************
package lab08_2023;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class CirclePanel extends JPanel {
   private final int WIDTH = 600, HEIGHT = 400;
   private Circle circle;

   //-----------------------------------------------------------------
   //  Sets up this panel to listen for mouse events.
   //-----------------------------------------------------------------
   public CirclePanel() {
      addMouseListener (new CirclesListener());
      setPreferredSize (new Dimension(WIDTH, HEIGHT));
   }

   //-----------------------------------------------------------------
   //  Draws the current circle, if any.
   //-----------------------------------------------------------------
   public void paintComponent (Graphics page) {
      super.paintComponent(page);
      if (circle != null) circle.draw(page);
   }

   //*****************************************************************
   //  Represents the listener for mouse events.
   //*****************************************************************
   private class CirclesListener implements MouseListener  {
      //--------------------------------------------------------------
      // Creates a new circle at the current location whenever the
      // mouse button is clicked and repaints.
      //--------------------------------------------------------------
      public void mouseClicked (MouseEvent event) {
         circle = new Circle(event.getPoint());
         repaint();
      }
      //--------------------------------------------------------------
      //  Provide empty definitions for unused event methods.
      //--------------------------------------------------------------
      public void mousePressed (MouseEvent event) {}
      public void mouseReleased (MouseEvent event) {}
      public void mouseEntered (MouseEvent event) {}
      public void mouseExited (MouseEvent event) {}
   }
}
```

Marking components:

Code correctness	90%
Clear comments	10%

Using the link at SCeLE, submit all files of your project (1 folder) and the corresponding *runnable JAR file*, zipped into a file with name: lab08_<class>_<TACode>_<yourName>_<yourNPM>.zip

Selamat Mengerjakan!

☺Happy Programming☺

L.Y.Stefanus & the Asdos Team