Graphical User Interface
Drawing

Object Oriented Programming
2024 First Semester
Shin-chi Tadaki (Saga University)

Today's theme

- Fundamentals of drawing in Java
- Handling mouse events
  - Mouse motions
  - Mouse button actions
- sample programs
  `https://github.com/oop-mc-saga/GUI2`

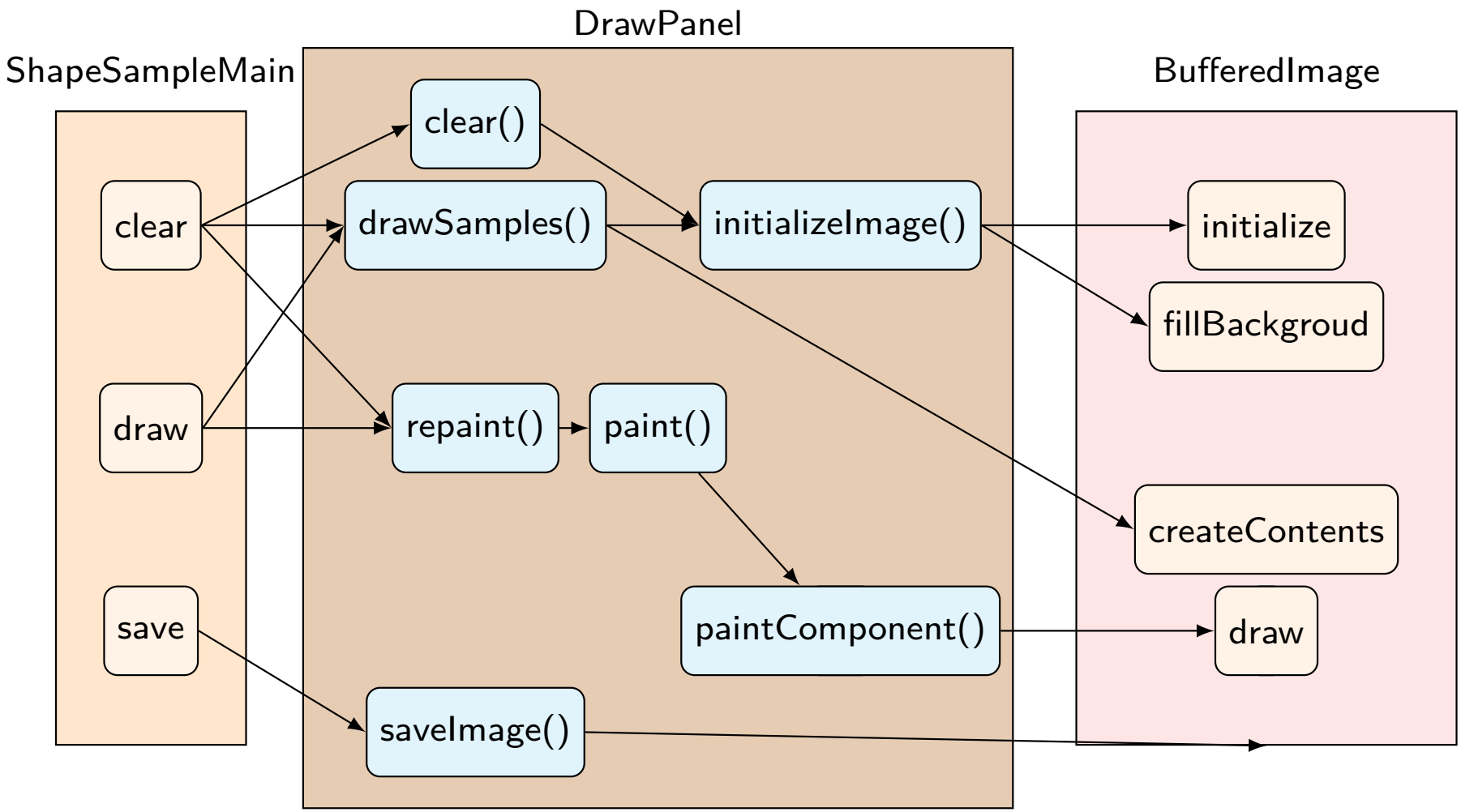
Drawing with `javax.swing`

- `paint()` calls the following methods sequentially
  - `paintComponent()`
  - `paintBorder()`
  - `paintChildren()`
- Override `paintComponent()` for your drawing purposes
  - Specify each drawing process
- Graphic contents are bound to an instance of `java.awt.Graphics`.

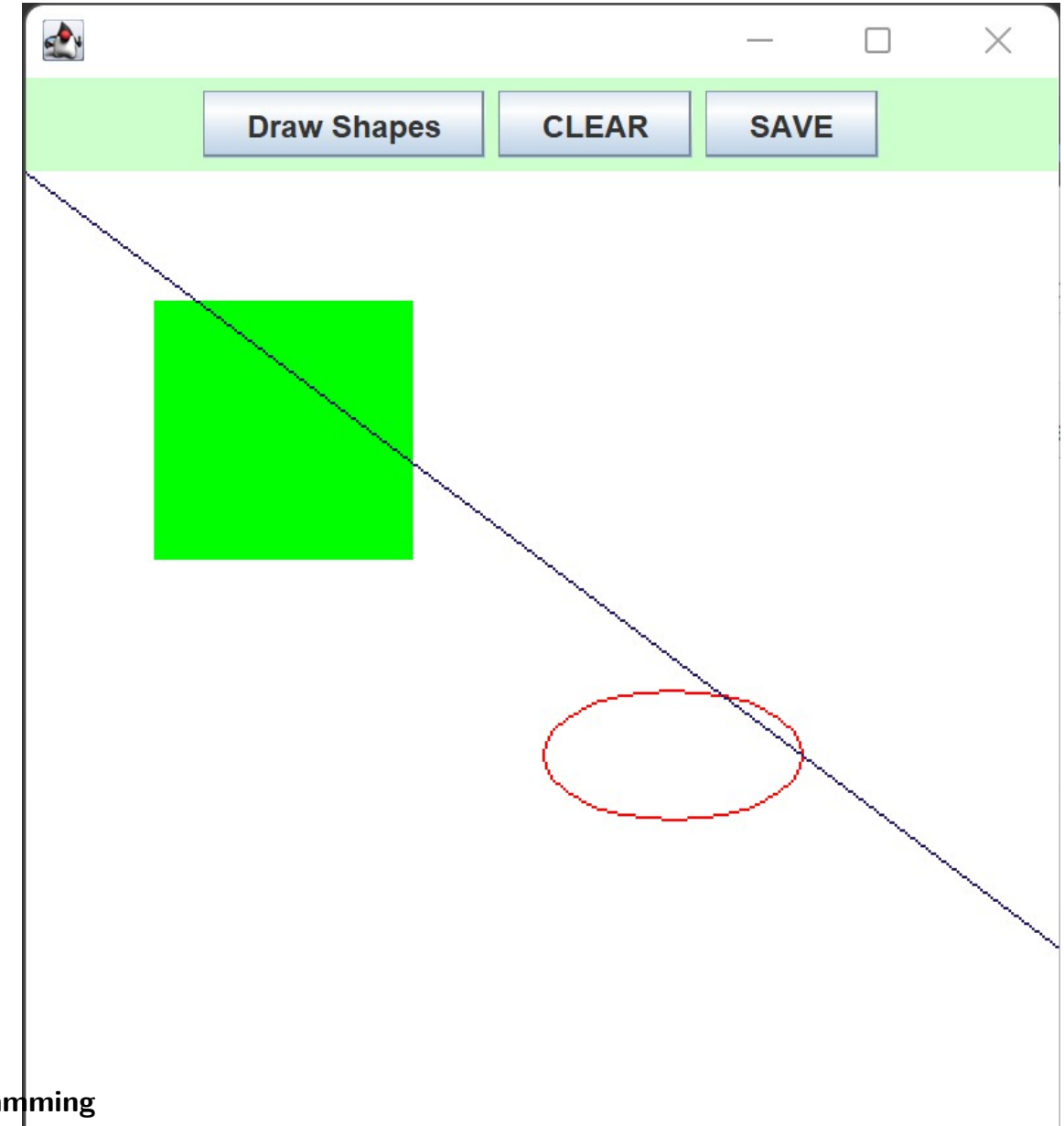
Redrawing processes

- Redrawing starts at, for instance,
  - changes in sequences of windows
  - restoring a window from icon
- It will need long time if drawing from scratch
- Redrawing event calls `repaint()`.
  - `repaint()` calls `paint()` internally
- Store image as `java.awt.image.BufferedImage` for quick redrawing
  - Put the stored image to canvas in `paintComponent()`
  - The stored image can be saved as a file

Components in shapeSample



Running screen



Initialize image

- Create a `BufferedImage` instance
- Fill the rectangle area of the image with the background color.
- `image.getGraphics()` returns the graphic contexts of `image`.

```
1  public void initializeImage() {
2      Dimension dimension = getPreferredSize();
3      //create new image
4      image = new BufferedImage(dimension.width, dimension.height,
5              BufferedImage.TYPE_INT_RGB);
6      Graphics2D g = (Graphics2D) image.getGraphics();
7      g.setColor(this.getBackground());//fill with background color
8      g.fillRect(0, 0, dimension.width, dimension.height);
9  }
```

Draw image

```
1  public void drawSamples() {
2      initializeImage();
3      Graphics2D g = (Graphics2D) image.getGraphics();
4      //rectangle
5      Rectangle2D.Double rect
6              = new Rectangle2D.Double(50., 50., 100., 100.);
7      g.setColor(Color.GREEN);
8      g.fill(rect);
9      //ellipse
10     Ellipse2D.Double ellipse
11             = new Ellipse2D.Double(200., 200., 100., 50.);
12     g.setColor(Color.RED);
13     g.draw(ellipse);
14     //straight line
15     g.setColor(new Color(30, 20, 100));
16     Line2D.Double line = new Line2D.Double(0., 0., 400., 300.);
17     g.draw(line);
18 }
```

Put image

- Put image in paintComponent()

```
1  public void paintComponent(java.awt.Graphics g) {
2      if (image == null) {return;}
3      ///put image
4      g.drawImage(image,
5          0, 0, image.getWidth(), image.getHeight(), this);
6  }
```

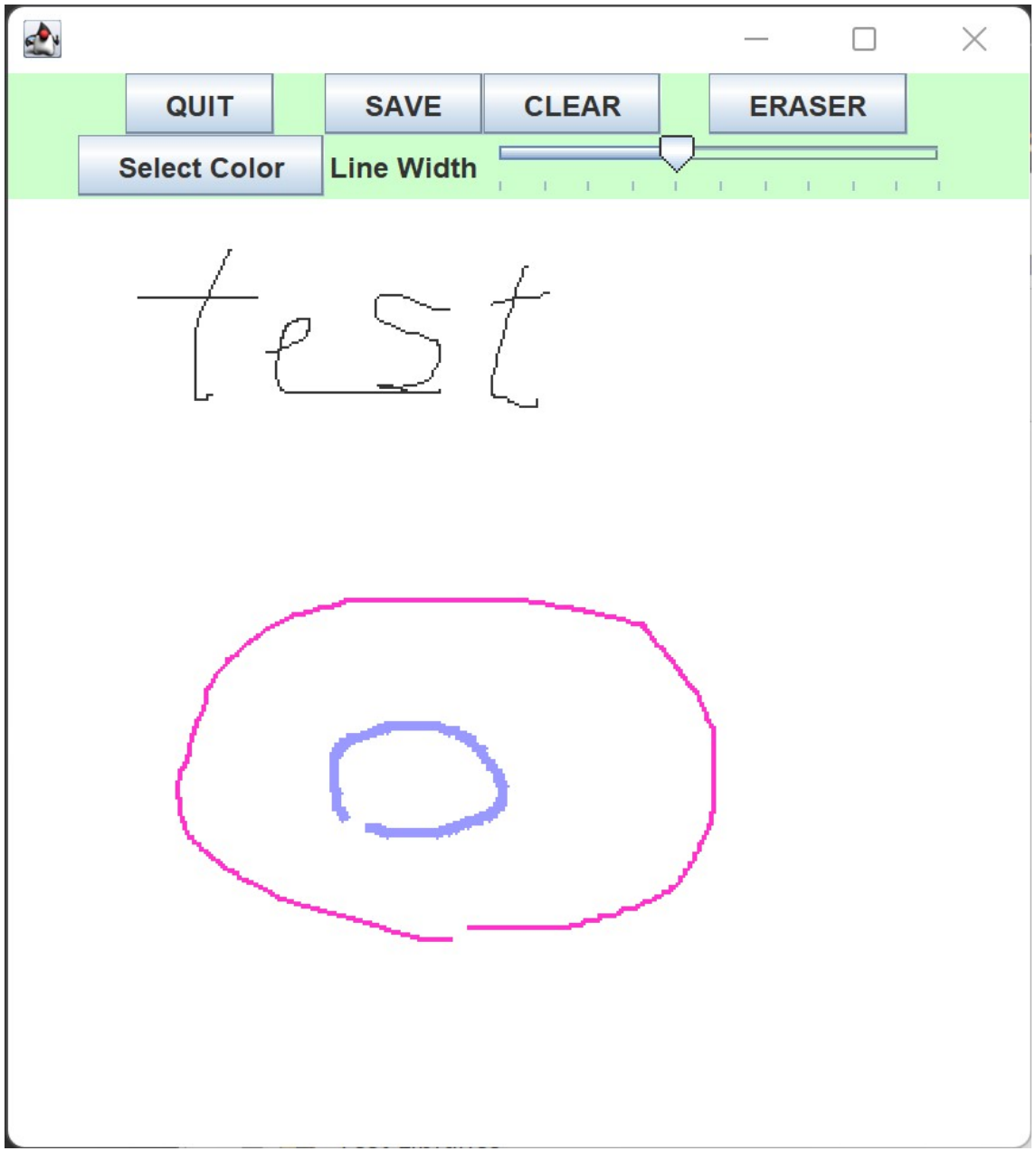
javax.imageio.ImageIO class

- Collection of methods for manipulating image files
- `read()`: reading image from file
- `write()`: writing image to file

```
1  public void saveImage(File file) {
2      if (!fileChooser.FileUtilGUI.checkWritable(file)) {
3          return;
4      }
5      try ( FileOutputStream out = new FileOutputStream(file)) {
6          String ext = FileIO.getExtention(file.getName());
7          try {
8              javax.imageio.ImageIO.write(image, ext, out);
9              String message
10                     = "Image is saved to " + file.getName();
11             fileChooser.FileUtilGUI.showMessage(message);
12         } catch (IOException ex) {
13             fileChooser.FileUtilGUI.showError(ex.getMessage());
14         }
15     }
16 }
```

Simple Drawer

- Draw curves using Mouse
  - `java.awt.event.MouseListener`
  - `java.awt.event.MouseMotionListener`
- Set line width
  - `java.awt.BasicStroke`
- Eraser
  - Draw thick curve with background color


QUIT
SAVE
CLEAR
ERASER
Select Color
Line Width
test

Handling mouse events

- Implement interfaces
  - `java.awt.event.MouseListener`
  - `java.awt.event.MouseMotionListener`
  - Implement methods for these interfaces
- Set listeners

```
1  addMouseListener(this);
2  addMouseMotionListener(this);
```

Drawing with mouse

- Press mouse button
  - `mousePressed()`
- Drag mouse
  - `mouseDragged()`
- Release mouse button
  - `mouseReleased()`

Basic concepts of drawing

- Press mouse button
  - Save mouse position to the `point` variable.
  - `java.awt.Point` holds integer (x, y) coordinate
- Drag mouse
  - Draw between the current and previous points
  - Save the current point to the `point` variable.
- Release mouse button
  - Draw between the current and previous points
  - Clear the `point` variable.

Connecting the current point to the previous

```
1  public void mouseDragged(MouseEvent e) {
2      if (point != null) {
3          Graphics2D g = (Graphics2D) image.getGraphics();
4          if (eraser) {//eraser case
5              g.setColor(this.getBackground());
6              g.setStroke(eraserStroke);
7          } else {
8              g.setColor(this.getForeground());
9              g.setStroke(stroke);
10         }
11         g.drawLine(point.x, point.y, e.getX(), e.getY());
12         point = new Point(e.getPoint());
13     }
14     repaint();
15 }
```

Mouse Methods

```
1  public void mousePressed(MouseEvent e) {
2      point = new Point(e.getPoint());
3  }
4
5  public void mouseReleased(MouseEvent e) {
6      if (point != null) {
7          Graphics2D g = (Graphics2D) image.getGraphics();
8          if (eraser) {//eraser case
9              g.setColor(this.getBackground());
10             g.setStroke(eraserStroke);
11         } else {
12             g.setColor(this.getForeground());
13             g.setStroke(stroke);
14         }
15         g.drawLine(point.x, point.y, e.getX(), e.getY());
16         point = null;
17     }
18     repaint();
19 }
```

Set line width

- Class for line properties
  - `java.awt.BasicStroke`
  - line width, terminal styles, etc.
- Set through `Graphic2D.setStroke()`

```
1  public void setLineWidth(int w) {
2      if (w < 1) {
3          w = 1;
4      }
5      stroke = new BasicStroke((float) w, BasicStroke.CAP_ROUND,
6              BasicStroke.JOIN_ROUND);
7  }
```

Set Color

- Using javax.swing.JColorChooser

```
1  private void selectColorActionPerformed(
2              java.awt.event.ActionEvent evt) {
3      java.awt.Color newColor =
4          JColorChooser.showDialog(
5              this, "Choose Foreground Color", getBackground());
6      drawPanel.setForeground(newColor);
7  }
```

Exercise: quiz

Let us add a polygon to `shapeSamples`.

Usage of `java.awt.geom.Path2D.Double` class

- `moveTo(double x, double y)` adds a point (x, y) to the path without drawing.
- `lineTo(double x, double y)` adds a point (x, y) to the path with drawing.
- `closePath()` closes the current subpath with drawing.