CIS2571 – Intro to Java

Chapter13 → Graphics

Information consolidated from a variety of textbook and online resources

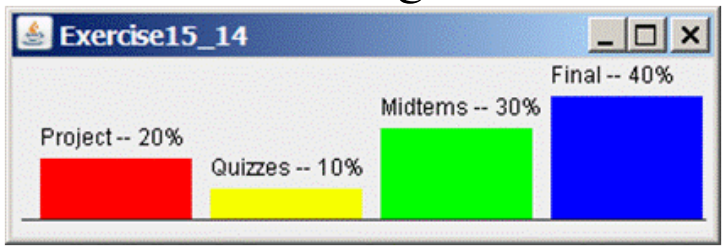
Topic Objectives

- Graphical Coordinate System
- Graphic Examples
  - Lines, Strings, Rectangles, Ovals
  - Arcs, Polygons
  - Text Centering
- Graphics Class
  - Case Study: FigurePanel Class*
- FontMetrics Class
  - Case Study: MessagePanel Class*
  - Case Study: StillClock Class*
- Displaying Images
  - Case Study: ImageViewer Class*

**case studies included for student review*

Graphics Class

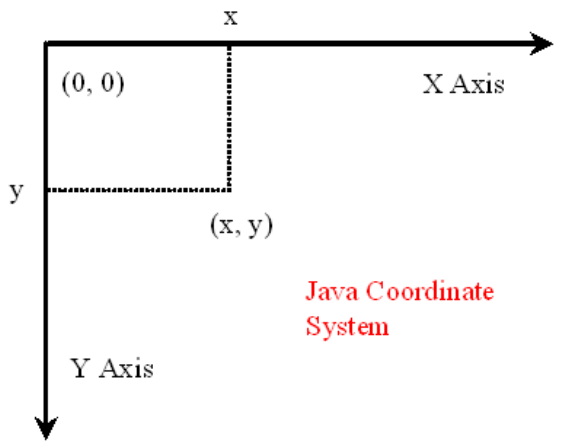
- Abstract base class for all graphics contexts to allow an application to draw device independent graphics onto components
  - Methods to draw strings, lines, rectangles, ovals, arcs, polygons, and images

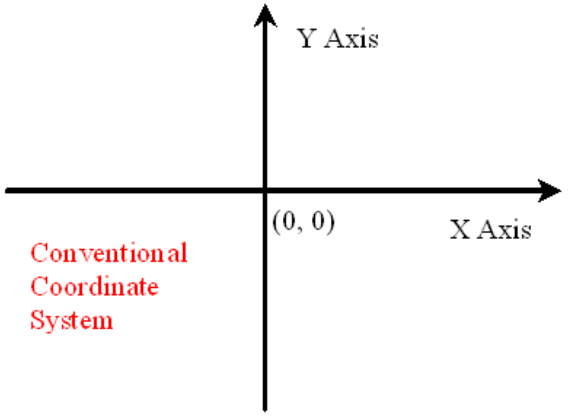


java.awt.Graphics	
+setColor(color: Color): void	Sets a new color for subsequent drawings.
+setFont(font: Font): void	Sets a new font for subsequent drwings.
+drawString(s: String, x: int, y: int): void	Draws a string starting at point (x, y).
+drawLine(x1: int, y1: int, x2: int, y2: int): void	Draws a line from (x1, y1) to (x2, y2).
+drawRect(x: int, y: int, w: int, h: int): void	Draws a rectangle with specified upper-left corner point at (x, y) and width w and height h.
+fillRect(x: int, y: int, w: int, h: int): void	Draws a filled rectangle with specified upper-left corner point at (x, y) and width w and height h.
+drawRoundRect(x: int, y: int, w: int, h: int, aw: int, ah: int): void	Draws a round-cornered rectangle with specified arc width aw and arc height ah.
+fillRoundRect(x: int, y: int, w: int, h: int, aw: int, ah: int): void	Draws a filled round-cornered rectangle with specified arc width aw and arc height ah.
+draw3DRect(x: int, y: int, w: int, h: int, raised: boolean): void	Draws a 3-D rectangle raised above the surface or sunk into the surface.
+fill3DRect(x: int, y: int, w: int, h: int, raised: boolean): void	Draws a filled 3-D rectangle raised above the surface or sunk into the surface.
+drawOval(x: int, y: int, w: int, h: int): void	Draws an oval bounded by the rectangle specified by the parameters x, y, w, and h.
+fillOval(x: int, y: int, w: int, h: int): void	Draws a filled oval bounded by the rectangle specified by the parameters x, y, w, and h.
+drawArc(x: int, y: int, w: int, h: int, startAngle: int, arcAngle: int): void	Draws an arc conceived as part of an oval bounded by the rectangle specified by the parameters x, y, w, and h.
+fillArc(x: int, y: int, w: int, h: int, startAngle: int, arcAngle: int): void	Draws a filled arc conceived as part of an oval bounded by the rectangle specified by the parameters x, y, w, and h.
+drawPolygon(xPoints: int[], yPoints: int[], nPoints: int): void	Draws a closed polygon defined by arrays of x and y coordinates. Each pair of (x[i], y[i]) coordinates is a point.
+fillPolygon(xPoints: int[], yPoints: int[], nPoints: int): void	Draws a filled polygon defined by arrays of x and y coordinates. Each pair of (x[i], y[i]) coordinates is a point.
+drawPolygon(g: Polygon): void	Draws a closed polygon defined by a Polygon object.
+fillPolygon(g: Polygon): void	Draws a filled polygon defined by a Polygon object.
+drawPolyline(xPoints: int[], yPoints: int[], nPoints: int): void	Draws a polyline defined by arrays of x and y coordinates. Each pair of (x[i], y[i]) coordinates is a point.

Graphical Coordinate System

- Need to know where to paint
- Each component has its own coordinate system with (0,0) at upper-left corner
  - x-coordinate increases to right
  - y-coordinate increases downward
  - measured in **pixels**
- Differs from conventional coordinate system





Graphics Class

- JVM creates Graphics object whenever a component is displayed
  - Invokes the paintComponent method of JComponent class
  - Created graphics object is passed to paintComponent method

```
protected void paintComponent(Graphics g)
```

- To draw on a component:
  - Define class to extend JPanel and
  - Override paintComponent method to specify what to draw
    - paintComponent method invoked when component is first displayed or whenever component needs to be redisplayed
    - User **should not** invoke paintComponent method directly
      - Invoking super.paintComponent necessary to ensure viewing area is cleared before new drawing is displayed
    - User should invoke the repaint() method (defined in Component class) to cause the paintComponent method to be called

See 13.1 TestPaintComponent.java

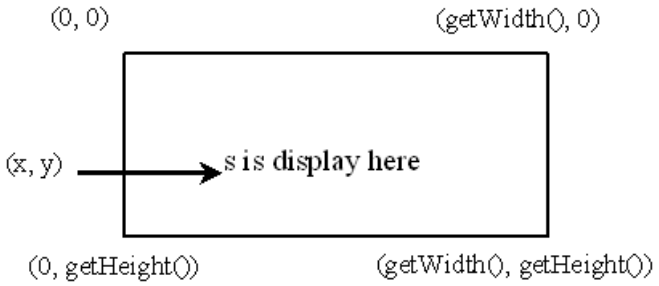
Graphics Class

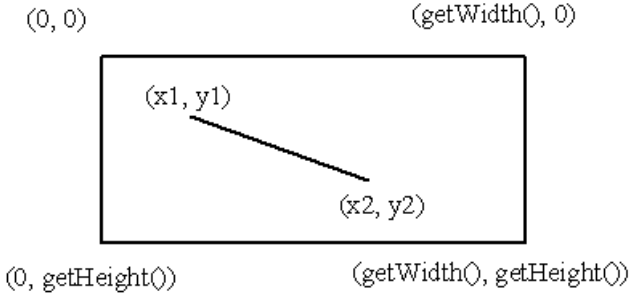
- Methods for
  - drawing strings

```
drawString(String s, int x, int y);
```

  - Lines

```
drawLine(int x1, int y1, int x2, int y2);
```

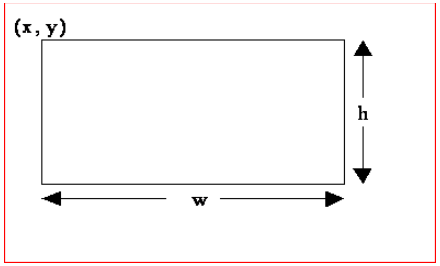


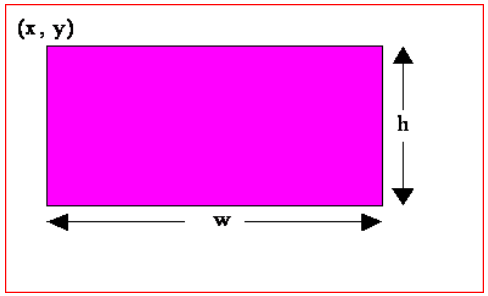


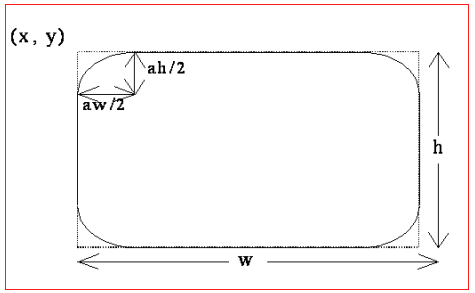
Graphics Class

- Methods for
  - rectangles

```
drawRect(int x, int y, int w, int h);
fillRect(int x, int y, int w, int h);
drawRoundRect(int x, int y, int w, int h, int aw, int
  ah);
fillRoundRect(int x, int y, int w, int h, int aw, int
  ah);
draw3DRect(int x, int y, int w, int h, boolean
  raised); // raised above or sunk into surface
fill3DRect(int x, int y, int w, int h, boolean
  raised); // raised above or sunk into surface
```







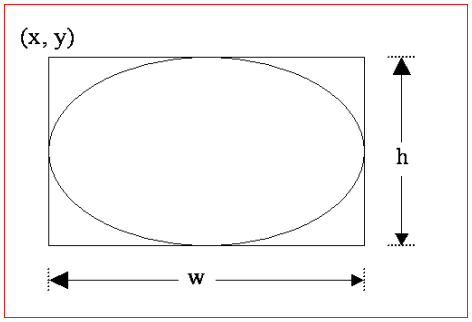
Graphics Class

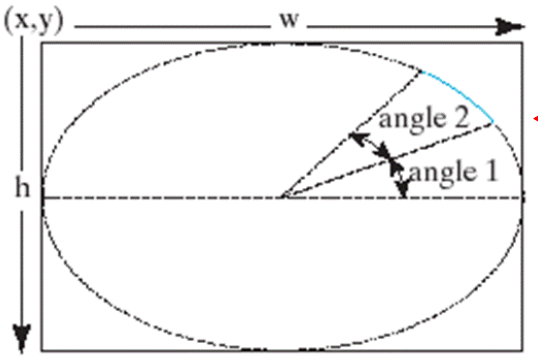
- Methods for
  - ovals

```
drawOval(int x, int y, int w, int h);
fillOval(int x, int y, int w, int h);
```

  - arcs

```
drawArc(int x, int y, int w, int h, int angle1, int
  angle2);
fillArc(int x, int y, int w, int h, int angle1, int
  angle2);
```





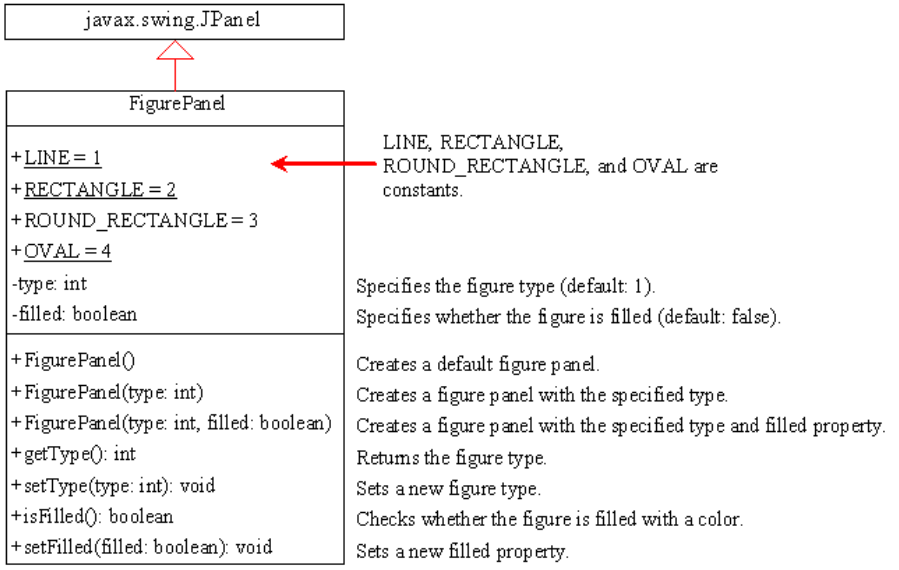
Angles are in degrees

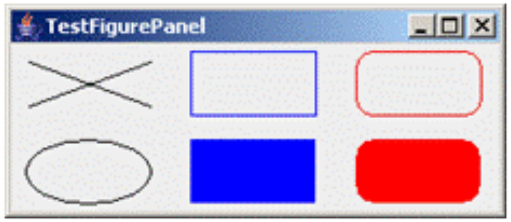
→Example

*case study for student review

Case Study: FigurePanel Class

- Class for displaying various figures
- Enables the user to set the figure type and specify whether the figure is filled, and displays the figure on a panel



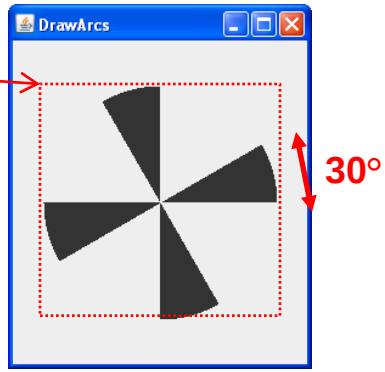


See 13.2 TestFigurePanel.java
See 13.3 FigurePanel.java

Graphics Class: Arcs Example

- Angles measured in degrees and follow typical mathematical conventions
  - 0 degrees is in easterly direction
  - Positive angles indicate counterclockwise rotation from easterly direction
  - Negative angles indicate clockwise rotation from easterly direction

(x, y)
x is xCenter - radius
y is yCenter - radius



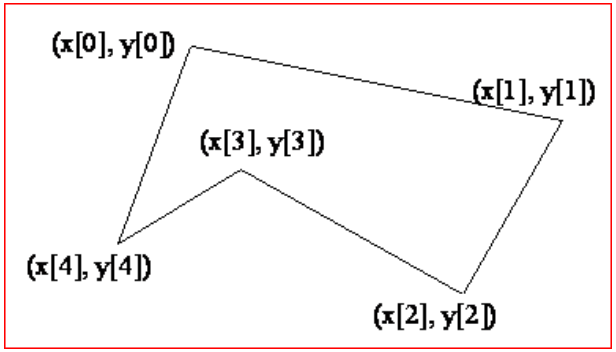
Angles	
Start	Span
0	30
90	30
180	30
270	30

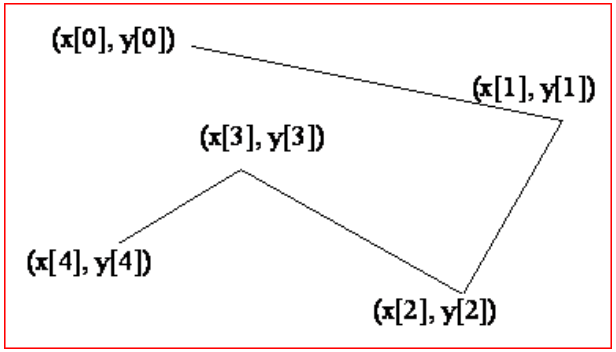
See 13.4 DrawArcs.java

Graphics Class

- Methods for
  - polygons

```
int[] x = {40, 70, 60, 45, 20};
int[] y = {20, 40, 80, 45, 60};
drawPolygon(x, y, x.length);
fillPolygon(x, y, x.length);
--OR--
Polygon p = new Polygon();
p.addPoint(40, 20);
p.addPoint(70, 40);
p.addPoint(60, 80);
p.addPoint(45, 45);
p.addPoint(20, 60);
drawPolygon(p);
fillPolygon(p);
```





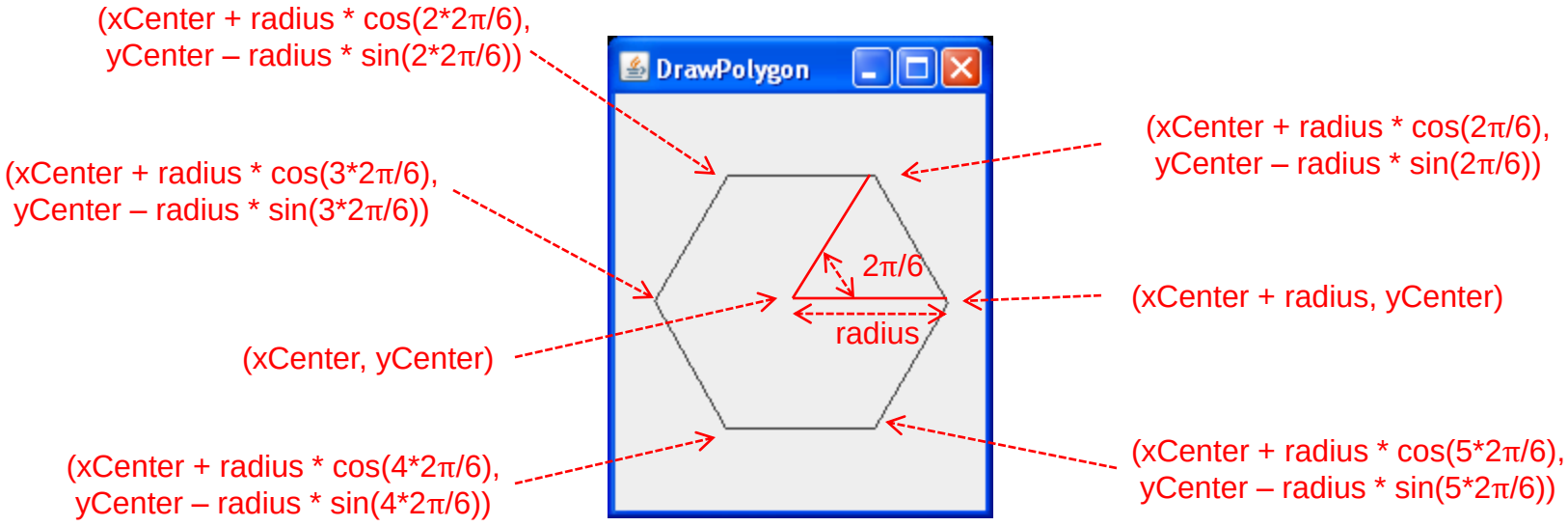
  - polylines

```
drawPolyline(x, y, x.length);
```

→Example

*case study for student review

Graphics Class: Draw Polygon Example



See 13.5 DrawPolygon.java

FontMetrics Class

- Abstract class used to display a string centered within a panel by using font width and height measurements

  - Leading → amount of space between lines of text
    `int getLeading()`
  - Ascent → distance from baseline to ascent line
    `int getAscent()`
  - Descent → distance from baseline to descent line
    `int getDescent()`
  - Height → sum of leading, ascent, and descent
    `int getHeight()`
  - Width → total string width (leftmost to rightmost) in this font
    `int stringWidth(String str)`

FontMetrics Class

- To get FontMetrics object for specific font, use Graphics class methods:

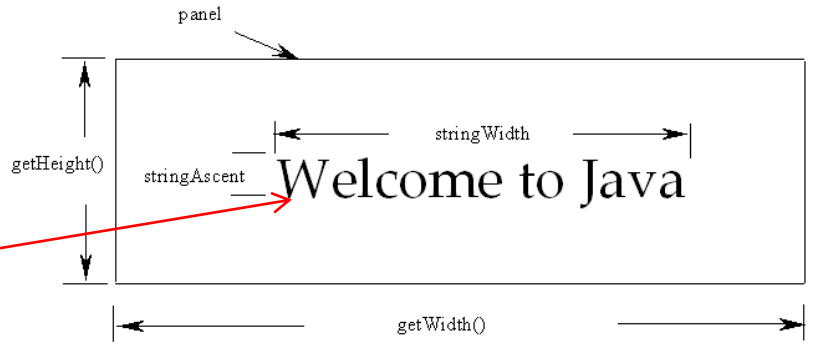
```
public FontMetrics getFontMetrics(Font font)
```

  - Returns font metrics of specified font

```
public FontMetrics getFontMetrics()
```

  - Returns font metrics of current font

- GraphicsEnvironment class describes available configurations such as Fonts



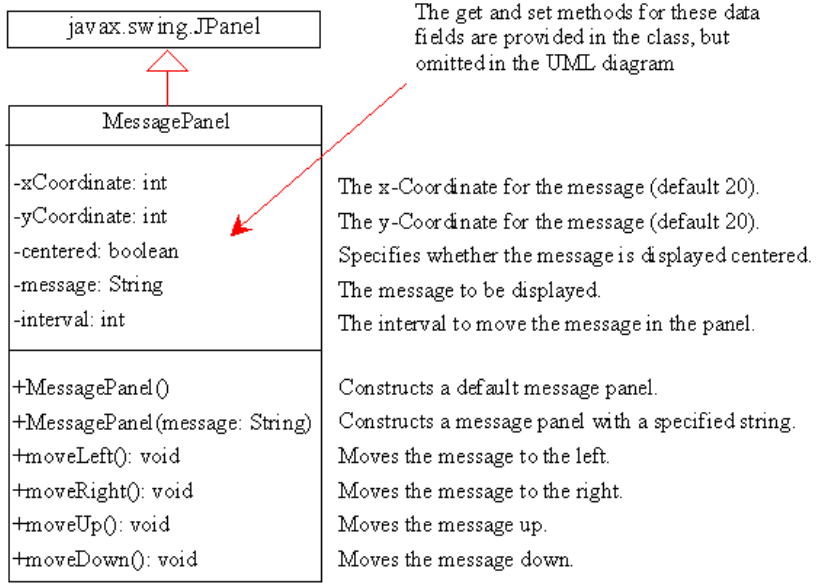
(xCoordinate, yCoordinate)
xCoordinate = getWidth() / 2 – stringWidth / 2;
yCoordinate = getHeight() / 2 + stringAscent / 2;

See 13.6 TestCenterMessage.java

*case study for student review

Case Study: MessagePanel Class

- Useful class that displays a message in a panel
- Enables user to set location of the message, center message, and move message with specified interval

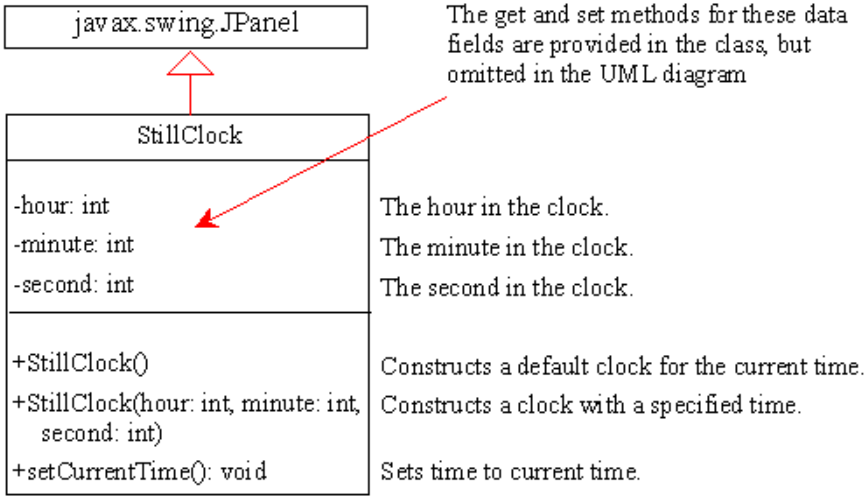


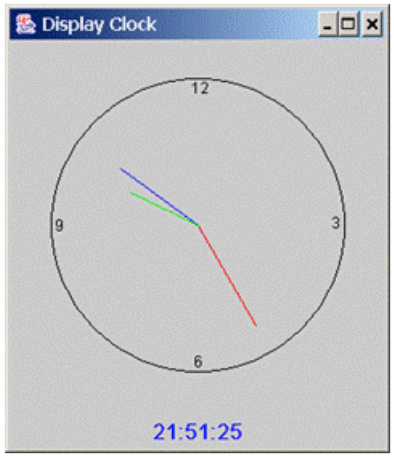
See 13.7 TestMessagePanel.java
See 13.8 MessagePanel.java

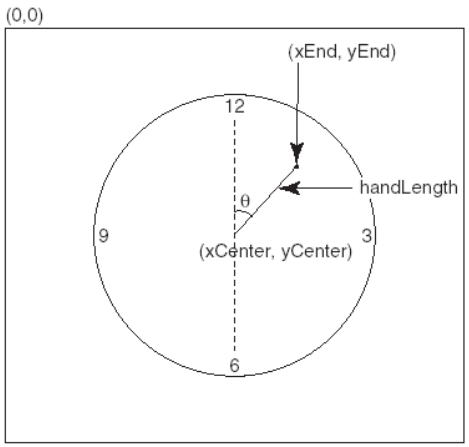
*case study for student review

Case Study: StillClock Class

- Display clock on panel
  - Uses circle and three hands for second, minute, and hour







See 13.9 DisplayClock.java
See 13.10 StillClock.java

Displaying Images

- Previously covered how to create **fixed size image icons** and display them in components

```
ImageIcon icon = new
  ImageIcon(getClass().getResource("image/us.gif"));
JLabel jlblImage = new Jlabel(imageIcon);
```

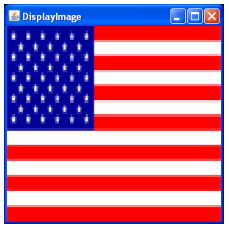
- **Flexible sized images** use java.awt.Image class

```
Image image = icon.getImage();
```

- Although label is simple way to display images, drawImage method of Graphics class is more flexible

ImageObserver specifies GUI component for receiving notifications of image information as the image is constructed

java.awt.Graphics	
+drawImage(image: Image, x: int, y: int, bgcolor: Color, observer: ImageObserver): void	Draws the image in a specified location. The image's top-left corner is at (x, y) in the graphics context's coordinate space. Transparent pixels in the image are drawn in the specified color bgcolor. The observer is the object on which the image is displayed. The image is cut off if it is larger than the area it is being drawn on.
+drawImage(image: Image, x: int, y: int, observer: ImageObserver): void	Same as the preceding method except that it does not specify a background color.
+drawImage(image: Image, x: int, y: int, width: int, height: int, observer: ImageObserver): void	Draws a scaled version of the image that can fill all of the available space in the specified rectangle.
+drawImage(image: Image, x: int, y: int, width: int, height: int, bgcolor: Color, observer: ImageObserver): void	Same as the preceding method except that it provides a solid background color behind the image being drawn.

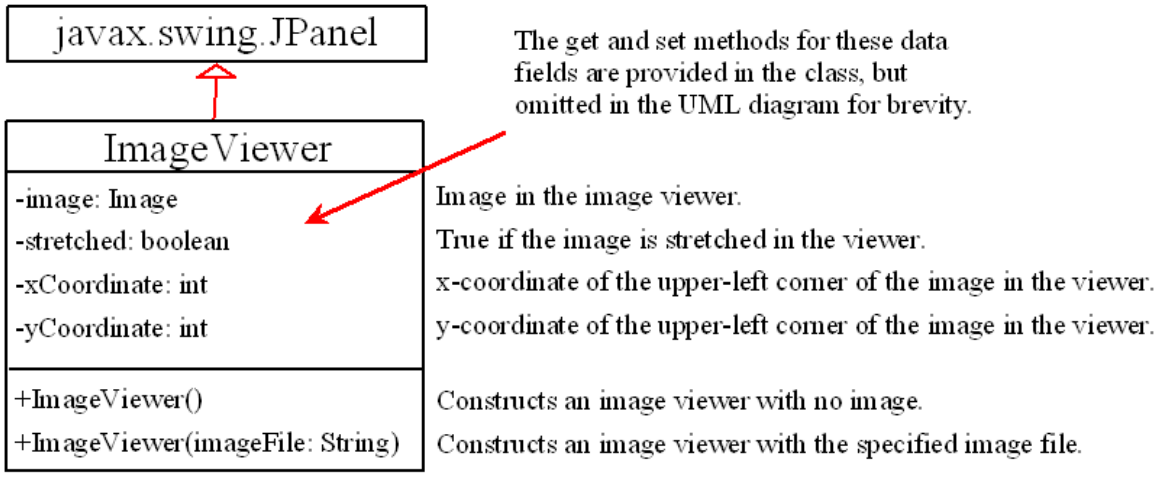


See 13.11 DisplayImage.java

*case study for student review

Case Study: ImageViewer Class

- reusable component named ImageViewer that displays a resizeable image in a panel



See 13.12 SixFlags.java
See 13.13 ImageViewer.java