CISC 3120: Graphics

Fall 2012

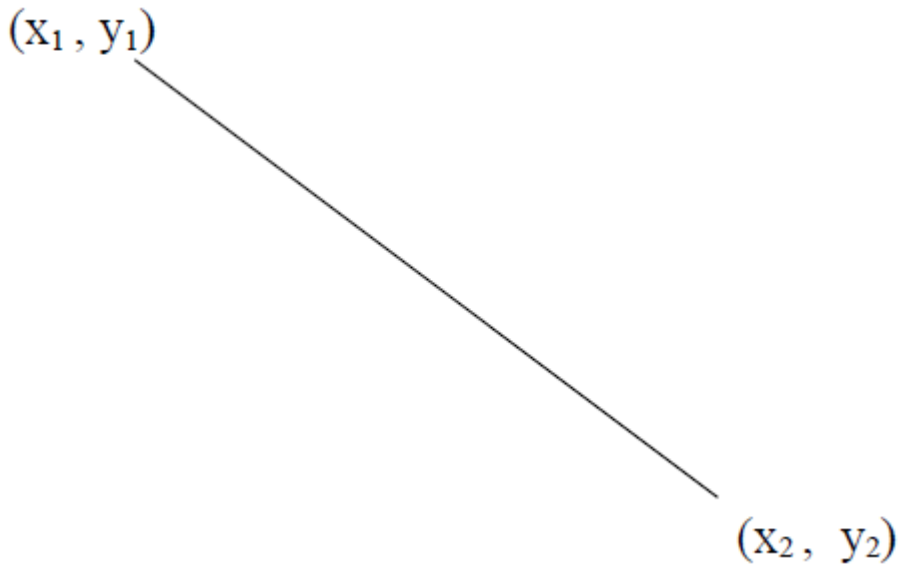
Zimi Li

Graphics

- java.awt.Graphics class
- X-windows coordinate system
- drawing primitives:
  - Lines
  - Rectangles
  - Ovals
  - Arcs
  - Polygons
- color

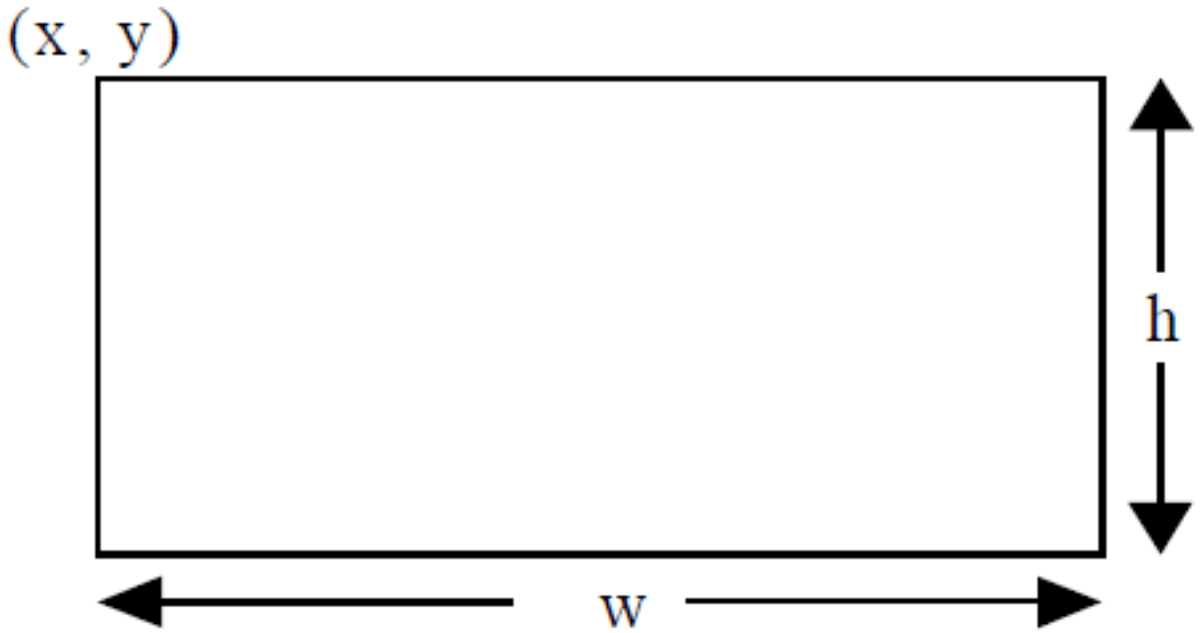
Drawing Lines

- `drawLine(x1, y1, x2, y2)`



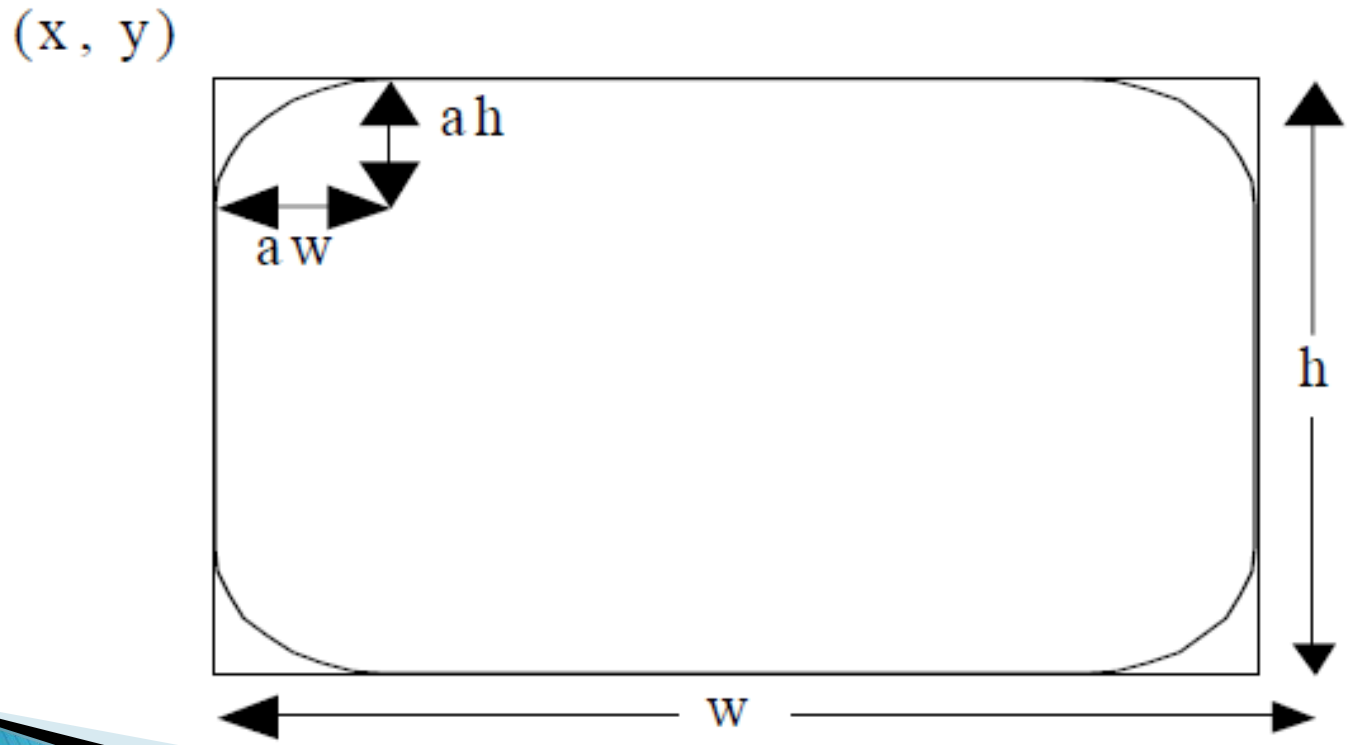
Drawing Rectangles

- `drawRect(x, y, w, h)`
- `fillRect(x, y, w, h)`



Drawing Rounded Rectangles

- `drawRoundRect(x, y, w, h, aw, ah)`
- `fillRoundRect(x, y, w, h, aw, ah)`

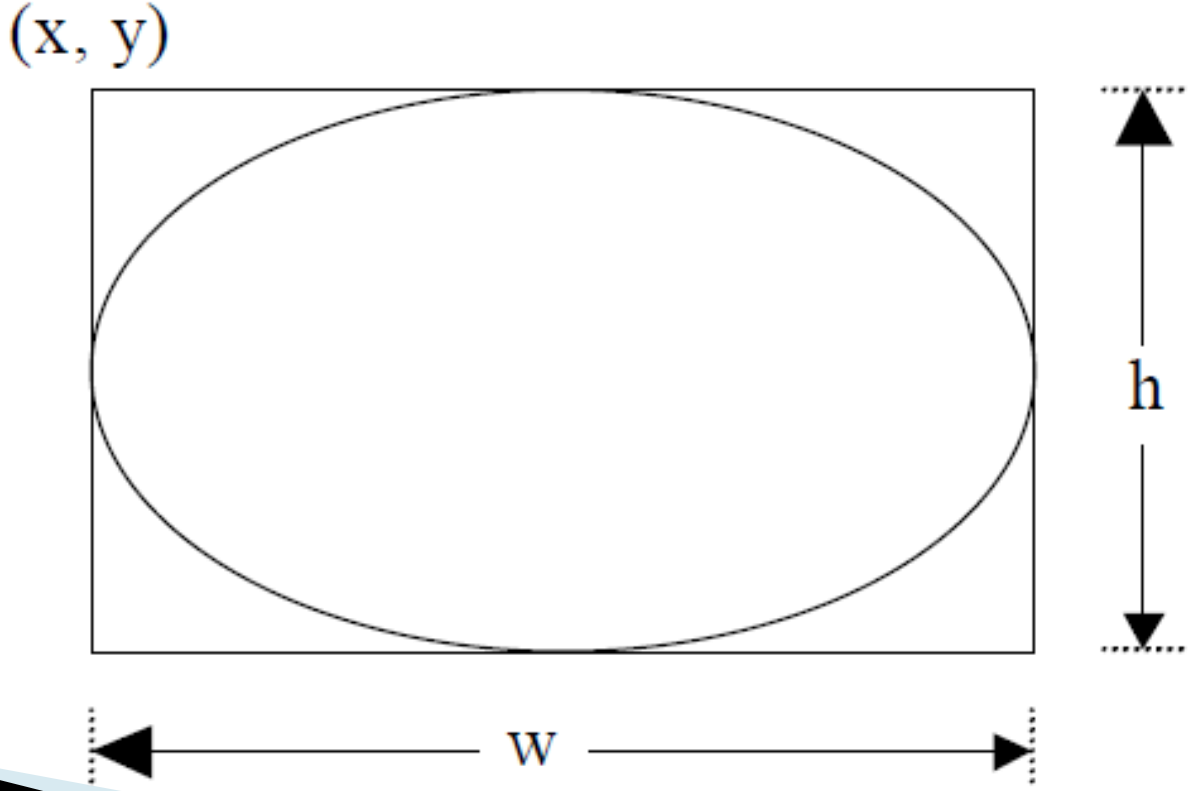


Draw Rectangles – Example

```
public void paintComponent(Graphics g) {
    super.paintComponent(g);
    // Set new color
    g.setColor(Color.red);
    // Draw a rectangle
    g.drawRect(30, 30, 100, 100);
    // Draw a rounded rectangle
    g.drawRoundRect(140, 30, 100, 100, 60, 30);
    // Change the color to cyan
    g.setColor(Color.cyan);
    // Draw a 3D rectangle
    g.fill3DRect(30, 140, 100, 100, true);
    // Draw a raised 3D rectangle
    g.fill3DRect(140, 140, 100, 100, false);
}
```

Drawing Ovals

- `drawOval(x, y, w, h)`
- `fillOval(x, y, w, h)`

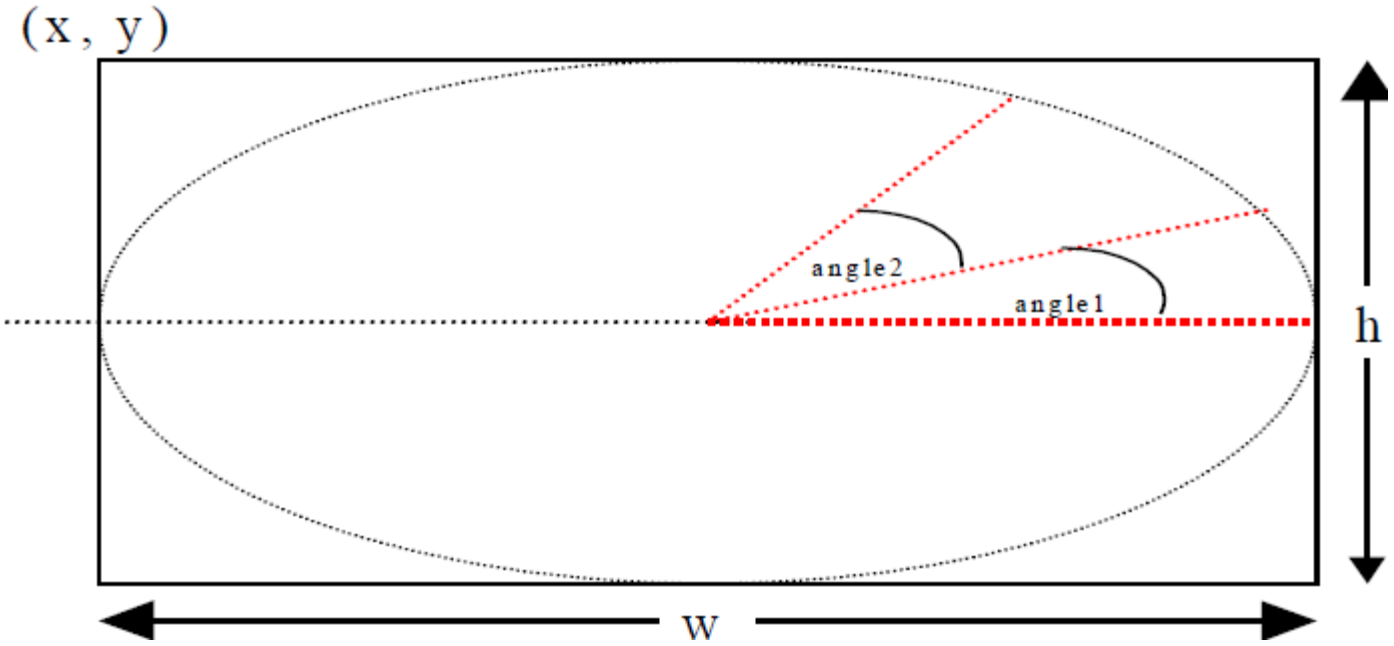


Drawing Ovals – Example

```
public void paintComponent(Graphics g) {
    super.paintComponents(g);
    g.drawOval(10, 30, 100, 60);
    g.drawOval(130, 30, 60, 60);
    g.setColor(Color.yellow);
    g.fillOval(10, 130, 100, 60);
    g.fillOval(130, 130, 60, 60);
}
```

Drawing Arcs

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

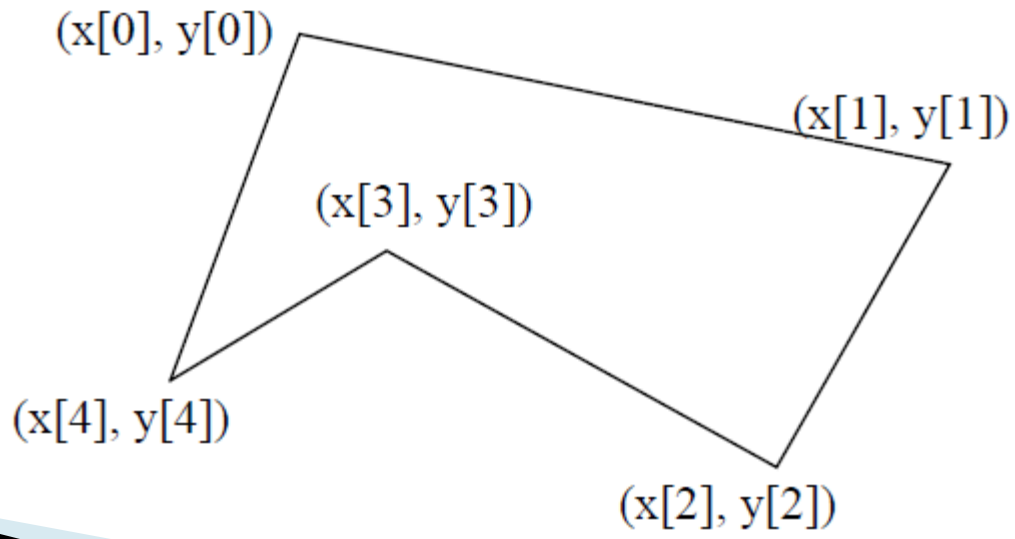


Drawing Arcs – Example

```
// Draw four blazes of a fan
public void paintComponent(Graphics g) {
    super.paintComponent(g);
    int xCenter = getWidth() / 2;
    int yCenter = getHeight() / 2;
    int radius = (int) (Math.min(getSize().width, getSize().height) * 0.4);
    int x = xCenter - radius;
    int y = yCenter - radius;
    g.fillArc(x, y, 2 * radius, 2 * radius, 0, 30);
    g.fillArc(x, y, 2 * radius, 2 * radius, 90, 30);
    g.fillArc(x, y, 2 * radius, 2 * radius, 180, 30);
    g.fillArc(x, y, 2 * radius, 2 * radius, 270, 30);
}
```

Drawing Polygons

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



Drawing Polygons – Example

```
super.paintComponent(g);
int xCenter = getWidth() / 2;
int yCenter = getHeight() / 2;
int radius = (int) (Math.min(getSize().width, getSize().height) * 0.4);
// Create a Polygon object
Polygon polygon = new Polygon();
polygon.addPoint(xCenter + radius, yCenter);
polygon.addPoint((int) (xCenter + radius * Math.cos(2 * Math.PI / 6)),
        (int) (yCenter - radius * Math.sin(2 * Math.PI / 6)));
polygon.addPoint(
        (int) (xCenter + radius * Math.cos(2 * 2 * Math.PI / 6)),
        (int) (yCenter - radius * Math.sin(2 * 2 * Math.PI / 6)));
polygon.addPoint(
        (int) (xCenter + radius * Math.cos(3 * 2 * Math.PI / 6)),
        (int) (yCenter - radius * Math.sin(3 * 2 * Math.PI / 6)));
polygon.addPoint(
        (int) (xCenter + radius * Math.cos(4 * 2 * Math.PI / 6)),
        (int) (yCenter - radius * Math.sin(4 * 2 * Math.PI / 6)));
polygon.addPoint(
        (int) (xCenter + radius * Math.cos(5 * 2 * Math.PI / 6)),
        (int) (yCenter - radius * Math.sin(5 * 2 * Math.PI / 6)));
// Draw the polygon
g.drawPolygon(polygon);
```

Fonts (1)

- Fonts in Java are defined using the `java.awt.Font` class
- You can see which fonts (by name) are available in your system by using the `java.awt.Graphics-Environment.getAllFonts()` method
- You can get information about the point size of the font, whether it is italic, bold or plain
- You will most likely want to get the size of a string that might be drawn with the current font. first you need to create a `FontMetrics` object, then you can call the `FontMetrics. stringWidth(String str)` method to find the width

Fonts (2)

- useful font properties available in `FontMetrics`:
  - Ascent – the distance from the font's baseline to the top of an alphanumeric character
  - Descent – the distance from the font's baseline to the bottom of an alphanumeric character
  - Leading — aka interline spacing; the logical amount of space to be reserved between the descent of one line of text and the ascent of the next line
  - Height – the distance between the baseline of adjacent lines of text; equal leading + ascent + descent
  - Advance — the distance from the leftmost point to the rightmost point on the string's baseline

Drawing a Clock

- Use drawing methods to draw a clock showing the specified hour, minute, and second in a frame.
  - DispalyClock

Images

- You can load images from a URL and draw them
- Load them `using java.applet.getImage(URL url)` for an applet or `java.awt.Toolkit.getImage(URL url)` for an application
- Draw them using `Graphics.drawImage()` — there are a number of versions of this method
- Note that an Image is a Java object unto itself, defined in the `java.awt` package
- Also note that URL is a Java object that must be instantiated prior to using either of the `getImage()` methods

Animation

- Computer animation is kind of like an old-fashioned flip book
- You need to draw the object(s) being animated repeatedly, in each new location
- Each time, you calculate the new position of the object(s) and redraw
- You can either redraw the entire scene
- Or you can only redraw the object(s) that are moving
- But in the second case, you need to “erase” the object first, then move it to its new location and redraw
- The erasing part can be tricky if the background is not solid