GUIs

Graphical User Interfaces

CS111 Computer Programming

Department of Computer Science
Wellesley College

What's an Applet?

A program that can run inside a window (like a browser window), an instance of the Applet or JApplet class.

What's an Applet?

A program that can run inside a window (like a browser window), an instance of the Applet or JApplet class.

The browser expects an applet to respond to these messages:

```
init()
start()
paint()
stop()
destroy()
```


Hello.java

Hello.class


Hello.html


Hello.java

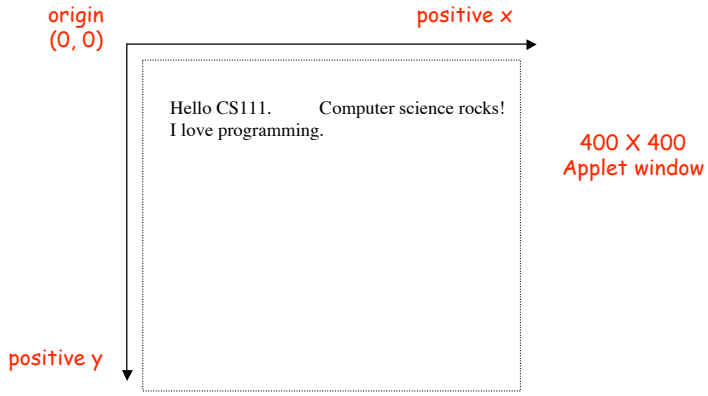
```
import java.awt.*;
import javax.swing.*;  // for JApplet
public class Hello extends JApplet
{
  public void paint (Graphics g)
  {
    g.drawString("Hello CS111", 25, 25);
    g.drawString("Computer science rocks!", 150, 25);
    g.drawString("I love programming.", 25, 50);
  }
}
```

Graphics object provided by Applet to which we send all drawing commands

Imagine 'g' is a painting machine. You're telling it what to paint and where. It does all the work.

Hello.html

```
<html>
  <body>
    <applet code="Hello.class" width=400 height=400></applet>
  </body>
</html>
```



A familiar face

```
import java.awt.*;
import javax.swing.*;  // for JApplet

public class Hello extends JApplet
{
    public void paint (Graphics g)
    {
        g.drawString("Hello CS111.", 25, 25);
        g.setColor(Color.orange); // Change font color to orange
        g.drawString("Computer science rocks!", 150, 25);
        g.setColor(Color.magenta); // Change font color to magenta
        g.drawString("I love programming.", 25, 50);
    }
}
```

The results

```
<html>
  <body>
    <applet code="Hello.class" width=400 height=400></applet>
  </body>
</html>
```

Graphics contract (partial)

`public void drawString(String str, int x, int y)`
Draws the text given by the specified string, using this graphics context's current font and color.

`public void setColor(Color c)`
Sets this graphics context's current color to the specified color.

`public void drawLine(int x1, int y1, int x2, int y2)`
Draws a line, using the current color, between the points (x1, y1) and (x2, y2) in this graphics context's coordinate system.

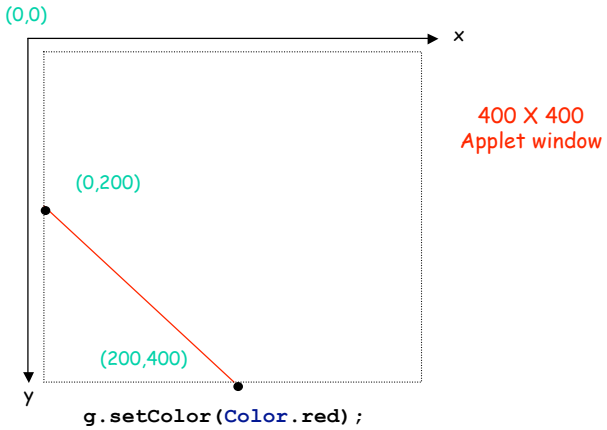
`public void drawRect(int x, int y, int width, int height)`
Draws the outline of the specified rectangle.

`public void fillRect(int x, int y, int width, int height)`
Fills the specified rectangle.

`public void drawOval(int x, int y, int width, int height)`
Draws the outline of an oval.

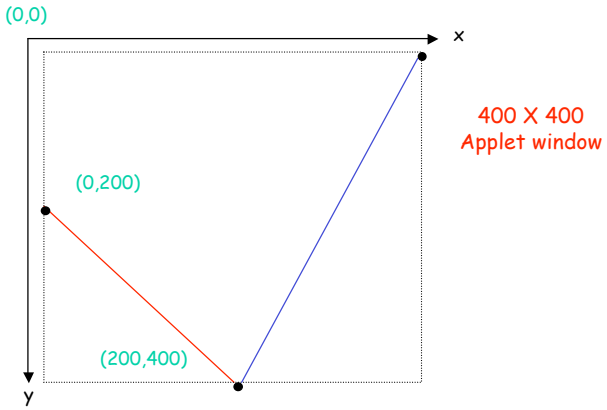
`public void fillOval(int x, int y, int width, int height)`
Fills the specified oval.

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



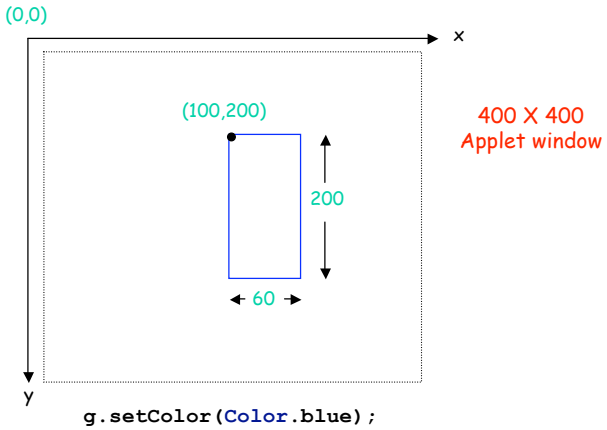

```
g.setColor(Color.red);
g.drawLine(0,200,200,400);
```


Look familiar?



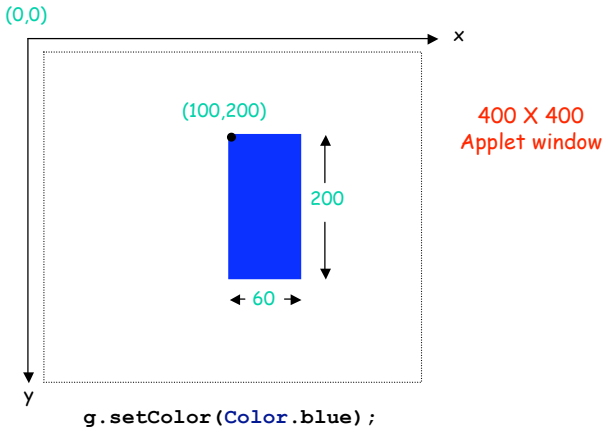

```
g.setColor(Color.red);
g.drawLine(0,200,200,400);
g.setColor(Color.blue);
g.drawLine(200,400,400,0);
```

drawRect(int x, int y, int width, int height)



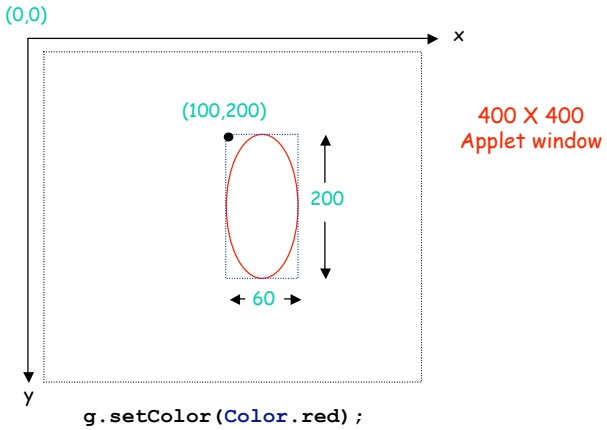

```
g.setColor(Color.blue);
g.drawRect(100,200,60,200);
```


fillRect(int x, int y, int width, int height)




```
g.setColor(Color.blue);
g.fillRect(100,200,60,200);
```

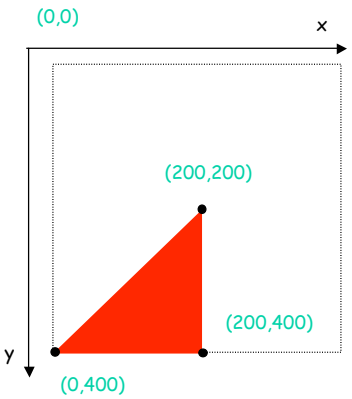
drawOval(int x, int y, int width, int height)




```
g.setColor(Color.red);
g.drawOval(100,200,60,200);
```


The Polygon class

- A member of the Polygon class has multiple vertices.
- We create a new Polygon with, you guessed it, a constructor, then add vertices.
- Polygons don't appear until asked.

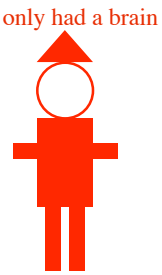


Creates an empty Poly

and Poly gets the point

```
Polygon anna = new Polygon();
anna.addPoint(0,400);
anna.addPoint(200,400);
anna.addPoint(200,200);
g.drawPolygon(anna);
g.fillPolygon(anna);
```

The Tin Man




```
g.setColor(Color.red);
g.drawOval(100,100,50,50);  // Head
Polygon hat = new Polygon();
hat.addPoint(100,100);
hat.addPoint(125,70);
hat.addPoint(150,100);
g.fillPolygon(hat);
g.drawString("If I only had a brain", 80, 60);
g.fillRect(100,150,50,80);  // Body
g.fillRect(105,230,15,50);  // Leg
g.fillRect(130,230,15,50);  // Leg
g.fillRect(75,175,100,20);  // Arms
```

*See notes page for GraphicsDemo.java, Circles.java, and Ovals.java

Getting gui


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

public class SimpleButton extends JApplet
{
    private JButton runButton;

    public void init()
    {
        runButton = new JButton("Click me");
        this.getContentPane().add(runButton, "South");
    }
}
```

init() is called automatically to initialize the applet

Hey, nothing happens when I click


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

public class ActionButton extends JApplet
     implements ActionListener
{
    private JButton runButton;

    public void init()
    {
        runButton = new JButton("Click me");
        runButton.addActionListener(this);
        this.getContentPane().add(runButton, "South");
    }

    public void actionPerformed(ActionEvent event)
    {
        runButton.setLabel("Run button, run!");
    }
}
```

Package containing ActionListener and ActionEvent

Implements the contract of an ActionListener

Register our interest with this button

Implement the ActionListener interface. This is what handles the event.

Oz never did give nothing to the Tin Man

```
class OzCanvas extends JPanel
{
     public void paintComponent(Graphics g)
     {
         g.setColor(Color.red);
         g.drawOval(100,100,50,50);  // Head
         Polygon hat = new Polygon();
         hat.addPoint(100,100);
         hat.addPoint(125,70);
         hat.addPoint(150,100);
         g.fillPolygon(hat);
         g.drawString("If I only had a brain", 80, 60);
         g.fillRect(100,150,50,80);  // Body
         g.fillRect(105,230,15,50);  // Leg
         g.fillRect(130,230,15,50);  // Leg
         g.fillRect(75,175,100,20);  // Arms
     }
}
```

That he didn't, didn't already have

```
public class TinmanButton extends JApplet implements ActionListener
{
     private final String draw = "Draw";
     private final String hide = "Hide";
     private JButton drawOrHideButton;
     private OzCanvas ozCanvas;

     public void init()
     {
          drawOrHideButton = new JButton(draw);      // Draw first, Hide later
          drawOrHideButton.addActionListener(this);  // register our interest
          ozCanvas = new OzCanvas();                 // A place to draw
          ozCanvas.setVisible(false);                // hide for now

          Container content = getContentPane();
          content.setLayout(new BorderLayout());
          content.add(drawOrHideButton, BorderLayout.SOUTH);
          content.add(ozCanvas, BorderLayout.CENTER);
     }

     public void actionPerformed(ActionEvent event)
     {
          boolean newlyVisible = !ozCanvas.isVisible();
          ozCanvas.setVisible(newlyVisible);
          drawOrHideButton.setLabel( newlyVisible ? hide : draw);
     }
}
```


And we don't need no stinkin' browser!

```
public class TinmanButton extends JApplet implements ActionListener
{     … as before
    private static JFrame appFrame;   // When running as application

    public static void main (String[] args)
    {
         appFrame = new JFrame("Frame for Tinman Applet");
         TinmanButton tinman = new TinmanButton();
         appFrame.setSize(725, 600);
         appFrame.getContentPane().add("Center", tinman);
         appFrame.addWindowListener(new AppFrameCloser(appFrame));

         tinman.init();
         appFrame.show();
    }
}
class AppFrameCloser extends WindowAdapter
{
    private JFrame frame;
    public AppFrameCloser(JFrame frame)
    {
         this.frame = frame;
    }
    public void windowClosing(WindowEvent e)
    {
         frame.dispose();
    }
}
```