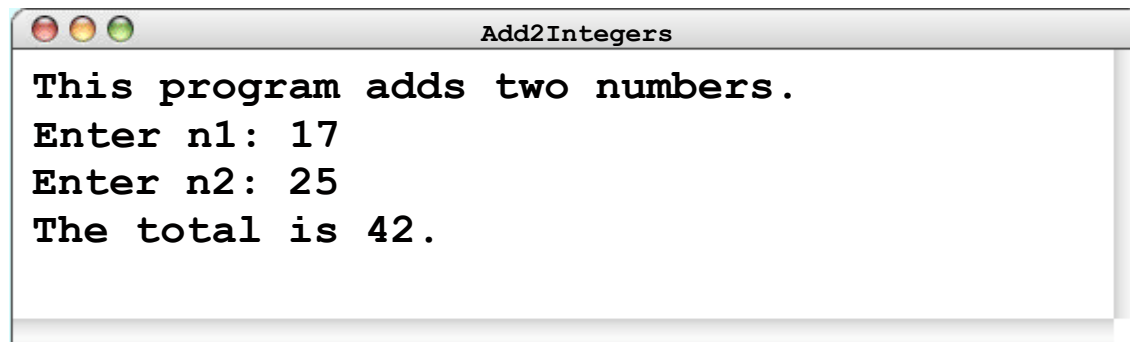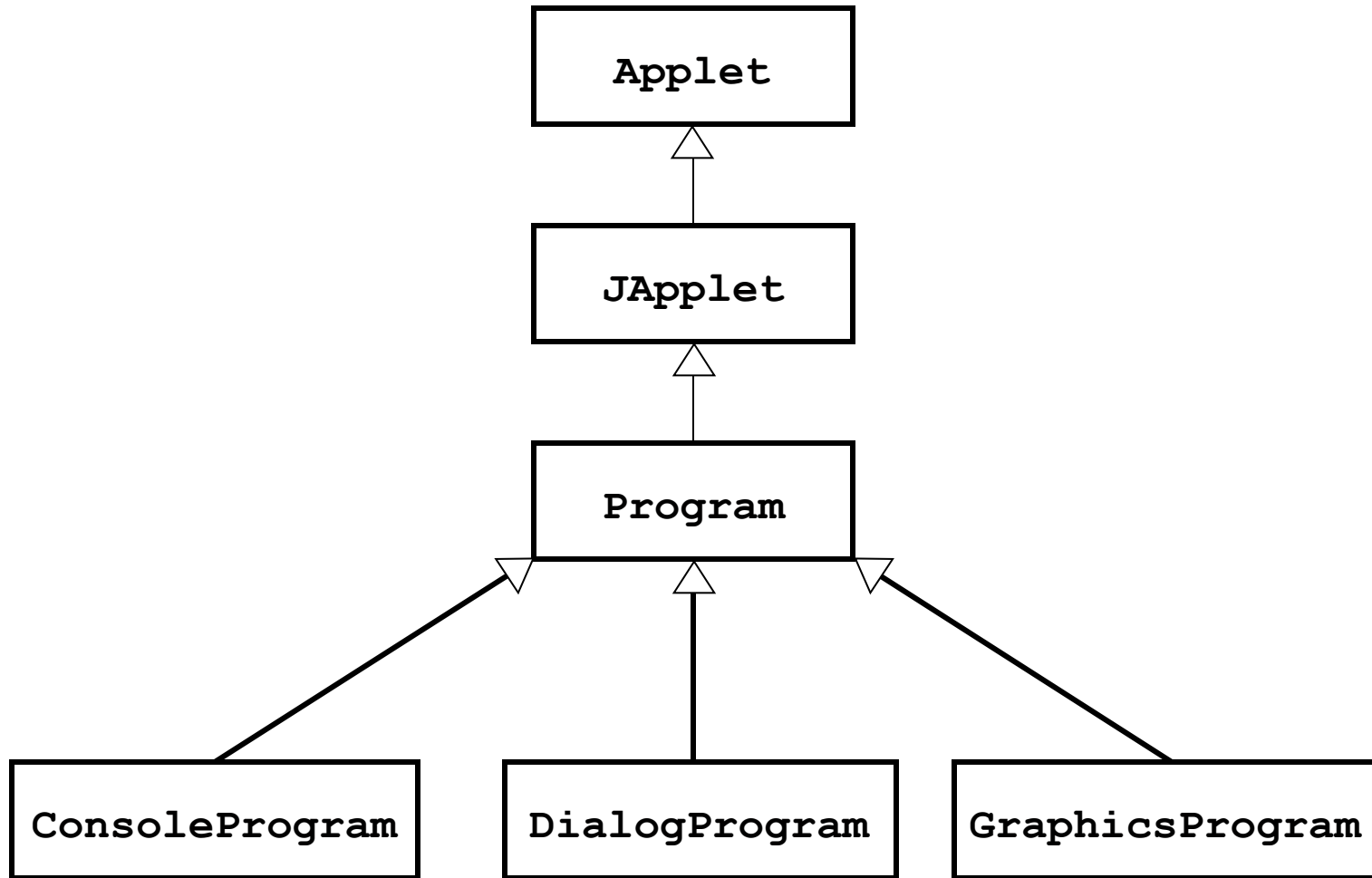# The `Add2Integers` Program

```
class Add2Integers extends ConsoleProgram {
    public void run() {
        println("This program adds two numbers.");
        int n1 = readInt("Enter n1: ");
        int n2 = readInt("Enter n2: ");
        int total = n1 + n2;
        println("The total is " + total + ".");
    }
}
```

n1	n2	total
17	25	42

```
● ● ●                    Add2Integers
This program adds two numbers.
Enter n1: 17
Enter n2: 25
The total is 42.
```

Graphic courtesy of Eric Roberts

acm.program Hierarchy

Graphic courtesy of Eric Roberts
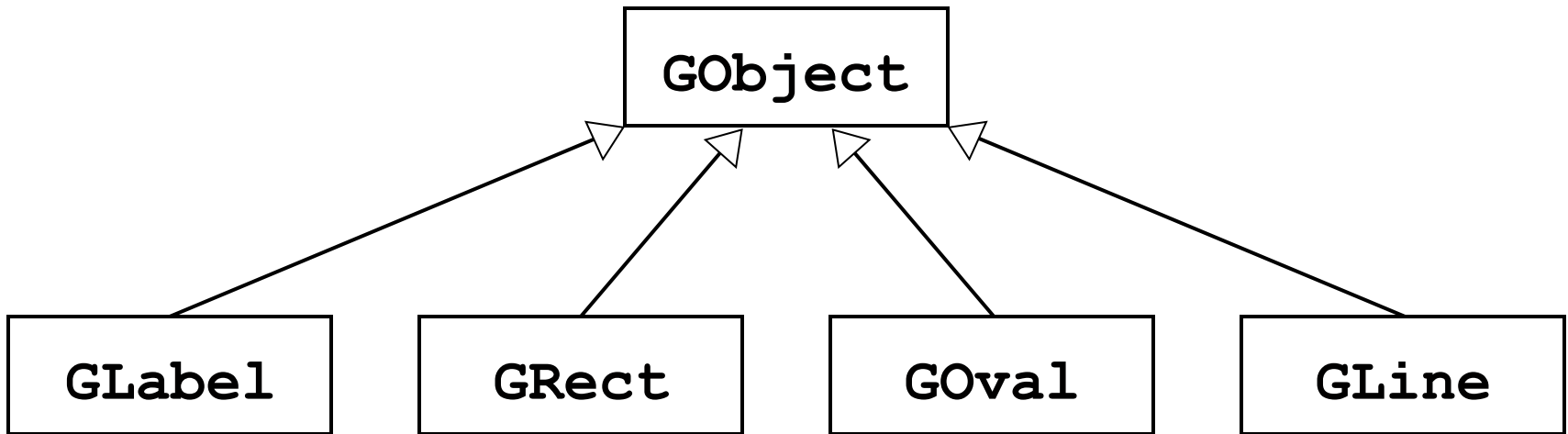
The `GObject` Hierarchy

The classes that represent graphical objects form a hierarchy, part of which looks like this:

```
                    ┌─────────────┐
                    │   GObject   │
                    └─────────────┘
         ↗         ↗        ↖         ↖
   ┌──────────┐ ┌──────────┐ ┌──────────┐ ┌──────────┐
   │  GLabel  │ │  GRect   │ │  GOval   │ │  GLine   │
   └──────────┘ └──────────┘ └──────────┘ └──────────┘
```

Sending Messages to a `GLabel`

The following program illustrates sending a message to an object. Note that the label doesn't appear until it is added to the canvas.

```
public class HelloProgram extends GraphicsProgram {
    public void run() {
        GLabel label = new GLabel("hello, world", 100, 75);
        label.setFont("SansSerif-36");
        label.setColor(Color.RED);
        add(label);
    }
}
```

label

hello, world

000 HelloProgram

hello, world
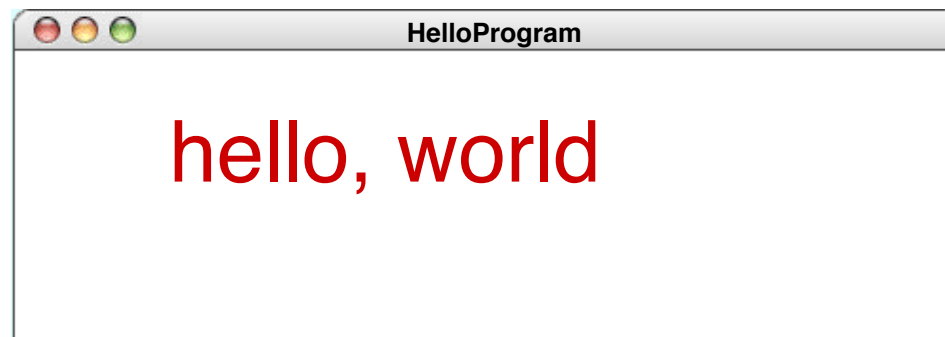
skip simulation

Graphic courtesy of Eric Roberts

Graphics Coordinates

- Origin is upper left

- Everything measured in pixels (dots on the screen)

- x coordinates increase to the right

- y coordinates increase going down

- `GLabel` coordinates are baseline of first character

(0, 0) →

HelloProgram

(100, 75) →

hello, world

Graphic courtesy of Eric Roberts
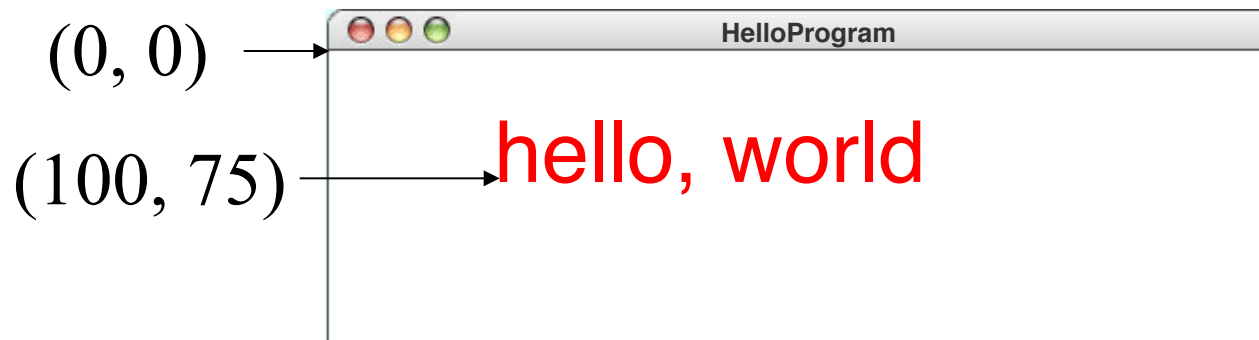
Operations on the `GObject` Class

The following operations apply to all **GObjects**:

---

object.**setColor(***color***)**
  Sets the color of the object to the specified color constant.

object.**setLocation(***x*, *y***)**
  Changes the location of the object to the point (*x*, *y*).

object.**move(***dx*, *dy***)**
  Moves the object on the screen by adding *dx* and *dy* to its current coordinates.

The standard color names are defined in the **java.awt** package:

```
Color.BLACK         Color.RED         Color.BLUE
Color.DARK_GRAY     Color.YELLOW      Color.MAGENTA
Color.GRAY          Color.GREEN       Color.ORANGE
Color.LIGHT_GRAY    Color.CYAN        Color.PINK
Color.WHITE
```

Based on slides by Eric Roberts

Operations on the `GLabel` Class

Constructor

> **new GLabel(***text***,** *x***,** *y***)**
>
> Creates a label containing the specified text that begins at the point (*x*, *y*).

Methods specific to the **GLabel** class

> *label*.**setFont(***font***)**
>
> Sets the font used to display the label as specified by the font string.

The font is typically specified as a string in the form

> **"***family-style-size***"**

family is the name of a font family
style is either **PLAIN**, **BOLD**, **ITALIC**, or **BOLDITALIC**
size is an integer indicating the point size

Drawing Geometrical Objects

Constructors

new GRect(*x* **,** *y* **,** *width* **,** *height* **)**
   Creates a rectangle whose upper left corner is at (x, y) of the specified size.

new GOval(*x* **,** *y* **,** *width* **,** *height* **)**
   Creates an oval that fits inside the rectangle with the same dimensions.

new GLine(x_0 **,** y_0 **,** x_1 **,** y_1 **)**
   Creates a line extending from (x_0, y_0) to (x_1, y_1).

Methods shared by the **GRect** and **GOval** classes

object.**setFilled(***fill***)**
   If *fill* is **true**, fills in the interior of the object; if **false**, shows only the outline.

object.**setFillColor(***color***)**
   Sets the color used to fill the interior, which can be different from the border.

Size of Graphics Window

Methods provided by **GraphicsProgram** class

getWidth()
    Returns the width of the graphics window.

getHeight()
    Returns the height of the graphics window.

Note: receiver of these calls is the **GraphicsProgram** itself, so we don't specify a separate object as receiver.