CS 106A, Lecture 12
More Graphics

reading:
Art & Science of Java, 9.4
You are here

Graphics Programs

The River of Java

Animation

Events

Memory

HW4: Breakout
Plan For Today

• Announcements
• Recap: Graphics
• GCompounds
• Getters
• Practice: Stoplights
• Practice: Checkerboard
Plan For Today

• Announcements
• Recap: Graphics
• GCompounds
• Getters
• Practice: Stoplights
• Practice: Checkerboard
Announcements: Docs

- Click the "Stanford Library Docs" link in the 106A website sidebar.
  - This site lists every kind of object in the Stanford libraries.
  - Click an object type on the left and see its behavior on the right.
  - These kinds of pages exist for Stanford libraries and standard Java.
Plan For Today

• Announcements
• Recap: Graphics
• GCompounds
• Getters
• Practice: Stoplights
• Practice: Checkerboard
The Graphics Canvas

0,0

\(\times\) 40,20

\(\times\) 120,40

\(\times\) 40,120
Collage Model
Graphical Objects

**GRect**
(x, y)

**GOval**
(x, y)

**GLine**
(x, y)

**GLabel**
Hello there!

**GImage**

**GGroundRect**

**GPolygon**

**GArc**
(x, y)
(x+w, y+h) (x+w, y+h) (x_1, y_1) (x_2, y_2)
Graphical Objects

GObject

GLabel  GRect  GOval  others...

GRect myRect = new GRect(50, 50, 350, 270);
## Primitives vs. Objects

<table>
<thead>
<tr>
<th>Primitive Variable Types</th>
<th>Object Variable Types</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>int</code></td>
<td><code>GRect</code></td>
</tr>
<tr>
<td><code>double</code></td>
<td><code>GOval</code></td>
</tr>
<tr>
<td><code>char</code></td>
<td><code>GLine</code></td>
</tr>
<tr>
<td><code>boolean</code></td>
<td><code>Scanner</code></td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

Object variables:
1. Have upper camel case types
2. You can call methods on them
3. Are constructed using `new`
Methods on Graphics Objects

We manipulate graphics objects by calling methods on them:

```
object.method(parameters);
```

Who?   What?   What specifically?

Example:

```
rect.setColor(Color.RED);
```
The following operations apply to all **GObjects**:

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object.setColor(color)</code></td>
<td>Sets the color of the object to the specified color constant.</td>
</tr>
<tr>
<td><code>object.setLocation(x, y)</code></td>
<td>Changes the location of the object to the point ((x, y)).</td>
</tr>
<tr>
<td><code>object.move(dx, dy)</code></td>
<td>Moves the object on the screen by adding (dx) and (dy) to its current coordinates.</td>
</tr>
<tr>
<td><code>object.getWidth()</code></td>
<td>Returns the width of the object</td>
</tr>
<tr>
<td><code>object.getHeight()</code></td>
<td>Returns the height of the object</td>
</tr>
</tbody>
</table>

and more…
new GRect(x, y, width, height);
− Creates a rectangle with the given width and height, whose upper-left corner is at (x, y)

new GRect(width, height);
− Same as above, but defaults to (x, y) = (0, 0)
new GOval(x, y, width, height);

- Creates an oval that fits inside a rectangle with the given width and height, and whose upper-left corner is at (x, y)

new GOval(width, height);

- Same as above, but defaults to (x, y) = (0, 0)
# GRect and GOval

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

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object.setFilled(fill)</code></td>
<td>If <code>fill</code> is <strong>true</strong>, fills in the interior of the object; if <strong>false</strong>, shows only the outline.</td>
</tr>
<tr>
<td><code>object.setFillColor(color)</code></td>
<td>Sets the color used to fill the interior, which can be different from the border.</td>
</tr>
<tr>
<td><code>object.setSize(width, height)</code></td>
<td>Sets the object’s size to be the given width and height</td>
</tr>
</tbody>
</table>
new GLine($x_0$, $y_0$, $x_1$, $y_1$);
- Creates a line extending from $(x_0, y_0)$ to $(x_1, y_1)$
new GLabel("your text here", x, y);
  – Creates a label with the given text, whose **baseline** starts at (x, y). NOT positioned according to the top-left corner!

new GLabel("your text here");
  – Same as above, but defaults to (x, y) = (0, 0)
# GLabel Methods

Methods specific to the **GLabel** class

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>label.getAscent()</code></td>
<td>Returns the height of the label above its baseline.</td>
</tr>
<tr>
<td><code>label.getDescent()</code></td>
<td>Returns the height of the label below its baseline.</td>
</tr>
<tr>
<td><code>label.setFont(font)</code></td>
<td>Sets the font used to display the label as specified by the font string.</td>
</tr>
</tbody>
</table>

The font is typically specified as a string in the form

```
"family-style-size"
```

*family* is the name of a font family (e.g. “SansSerif”)  
*style* is either **PLAIN**, **BOLD**, **ITALIC**, or **BOLDITALIC**  
*size* is an integer indicating the point size

Graphic courtesy of Eric Roberts
new GImage("your filename here", x, y);

– Creates a an image displaying the given file, whose upper-left corner is at (x, y)

new GImage("your filename here");

– Same as above, but defaults to (x, y) = (0, 0)
GImage Methods

`object.setSize(width, height)`

Sets the object’s size to be the given width and height
### GraphicsProgram Methods

- **GraphicsProgram** contains these useful methods:

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>add(obj);</code></td>
<td>adds a graphical object to the window</td>
</tr>
<tr>
<td><code>add(obj, x, y);</code></td>
<td>adds a graphical object to the window at position (x, y)</td>
</tr>
<tr>
<td><code>getElementAt(x, y)</code></td>
<td>return the object at the given (x, y) position(s)</td>
</tr>
<tr>
<td><code>getElementCount()</code></td>
<td>return number of graphical objects onscreen</td>
</tr>
<tr>
<td><code>getWidth(), getHeight()</code></td>
<td>return dimensions of window</td>
</tr>
<tr>
<td><code>remove(obj);</code></td>
<td>removes a graphical object from the window</td>
</tr>
<tr>
<td><code>removeAll();</code></td>
<td>remove all graphical objects from window</td>
</tr>
<tr>
<td><code>setCanvasSize(w, h);</code></td>
<td>set size of drawing area</td>
</tr>
<tr>
<td><code>setBackground(color);</code></td>
<td>set window's background color</td>
</tr>
</tbody>
</table>
Recap Practice: Centering
Recap Practice: Centering

```
getWidth() / 2.0
```

rectangle's x value = `getWidth() / 2.0 - W / 2.0`
Recap Practice: Centering

rectangle's y value = getHeight() / 2.0 - H / 2.0
Plan For Today

• Announcements
• Recap: Graphics
• GCompounds
• Getters
• Practice: Stoplights
• Practice: Checkerboard
A **GCompound** contains other GObjects. It’s useful when you want to do one operation on multiple GObjects at the same time.

```java
GCompound compound = new GCompound();
compound.add(shape);
compound.add(shape);
...
compound.add(shape);
add(compound);
```

– E.g. you can make a GCompound to represent a car.
setBackgroundColor(Color.YELLOW);
GCompound car = new GCompound();

GRect body = new GRect(10, 30, 100, 50);
body.setFilled(true);
body.setFillColor(Color.BLUE);
car.add(body);

GOval wheel1 = new GOval(20, 70, 20, 20);
wheel1.setFilled(true);
wheel1.setFillColor(Color.RED);
car.add(wheel1);

GOval wheel2 = new GOval(80, 70, 20, 20);
wheel2.setFilled(true);
wheel2.setFillColor(Color.RED);
car.add(wheel2);

GRect windshield = new GRect(80, 40, 30, 20);
windshield.setFilled(true);
windshield.setFillColor(Color.CYAN);
car.add(windshield);
add(car);  // at 0,0! Where we want this “sub-canvas” to go
Plan For Today

• Announcements
• Recap: Graphics
• GCompounds
• Getters
• Practice: Stoplights
• Practice: Checkerboard
Methods of graphical objects that return values:

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>obj.getColor()</code></td>
<td>the color used to color the shape outline</td>
</tr>
<tr>
<td><code>obj.getFillColor()</code></td>
<td>the color used to color the shape interior</td>
</tr>
<tr>
<td><code>obj.getX()</code></td>
<td>the left x-coordinate of the shape</td>
</tr>
<tr>
<td><code>obj.getY()</code></td>
<td>the top y-coordinate of the shape</td>
</tr>
<tr>
<td><code>obj.getWidth()</code></td>
<td>number of pixels wide the shape is</td>
</tr>
<tr>
<td><code>obj.getHeight()</code></td>
<td>number of pixels tall the shape is</td>
</tr>
</tbody>
</table>

Example: Swapping the x/y coordinates of a shape:

```java
GRect rect = new GRect(...);
...
int rx = rect.getX();
int ry = rect.getY();
rect.setLocation(ry, rx);
```
Plan For Today

• Announcements
• Recap: Graphics
• GCompounds
• Getters

• Practice: Stoplights
• Practice: Checkerboard
Practice: Stoplights

How would you make a method for drawing stoplights of different locations and sizes?
Practice: Stoplights

What information do we need in order to draw this?
Write a graphical program **LineArt** that draws a series of lines (see lecture code for solution):

- Outer square is at (10, 30) and size 200x200
- each line is 10px apart in each dimension

**coordinates of top-left lines:**
- (210, 30) to (10, 30)
- (200, 30) to (10, 40)
- (190, 30) to (10, 50)
- ...
- (20, 30) to (10, 220)

**coordinates of bottom-right lines:**
- (210, 30) to (210, 230)
- (210, 40) to (200, 230)
- ...
- (210, 220) to (20, 230)
Plan For Today

• Announcements
• Recap: Graphics
• GCompounds
• Getters
• Practice: Stoplights
• Practice: Checkerboard
Write a graphical program named **Checkerboard** that draws a checkerboard pattern using GRects.
Milestone 2
Milestone 3
Milestone 3

- Notice the pattern if we add the row and column indexes...

![Diagram](image.png)
Recap

• Announcements
• Recap: Graphics
• GCompounds
• Getters
• Practice: Stoplights
• Practice: Checkerboard

Next time: Animation