Graphics in Java

10-28-2004

Opening Discussion

- Do you have any questions about the quiz?
- What did we talk about last class? Do you have any code to show?
- Do you have any questions about the assignment? Remember that the design for assignment #5 is due today.

Drawing on Your Own

- Sometimes you want to have more control than just using the GUI components provided by the Java libraries. In these situations you might want to have custom control over what gets drawn to the space inhabited by a given component.
- The power and quality of this was enhanced with the Graphics 2D library.

Overriding the paint Method

- The way that you can control what is drawn on a component is to override the paint method of that component (for Swing use paintComponent). This means that you need to create a subclass of a component. I typically do custom components from JPanels.
- The paint method takes a java.awt.Graphics object. What you draw with that object shows up on the

The Graphics Class

- The Graphics class encapsulates basic drawing operations with the data that tells how they should be drawn.
- There are methods to draw, or fill, basic shapes (lines, ovals, rectangles, polygons) as well as images and strings.
- A variety of set methods tell it how to draw those things.

The Graphics2D Class

- If you are using Swing, the Graphics object will actually be an instance of Graphics2D. We'll talk more about the power of this next class, but with it, you will typically draw with the draw(Shape s) and fill(Shape s) methods.
- Shape is very powerful and flexible, but they provide some standard shapes in the java.awt.geom package.

Clipping

- The process of clipping is really setting what part of a surface you will draw to. You can set a rectangular region or a shape and anything that would be drawn outside of that region doesn't get drawn.
- Normally the clipping region is set to be the size of the component. You can set it smaller to make sure things are located to a smaller region.

Colors

- You can set what color to draw with. The way you do this is by giving the Graphics object a Color object. With Graphics2D you should instead use setPaint. Color is a subclass of Paint.
- There are a number of predefined Color objects for standard colors. You can also define arbitrary colors using constructors that take red, green, and blue values.

Fonts

- You can also set what font you want the Graphics object to use when you draw text to it.
- The potential complexity rises a fair bit here, but you can easily create Font objects by providing a standard font name, style, and size (see API for details).
- Finding the size of the strings drawn in the given font requires using LineMetrics

Code

- Let’s look at and write some code to create a JPanel that does custom drawing and has some type of interaction with the user.

More Advanced Graphics

- Next class we will continue to discuss the more advanced features of Java2D.
- There is also an option Java package called Java3D that allows you to do 3D graphics fairly easily. You can look on the web for details though teaching it is beyond the scope of this class. It also makes code less portable because it is not standard and required either OpenGL or DirectX to run.

Minute Essay

- What did we talk about today? What do you think about the project and how it is going so far? Is having the large project helping or hindering your ability to learn about object-orientation?