

# **Java Graphics 2**



**3-24-2005**

# Opening Discussion



- What did we talk about last class? Do you have any code to show?
- Do you have any questions about the assignment?
- Mock programming competition and extra credit.

# Capabilities of Graphics2D



- The Java2D libraries add significant capabilities to the graphics package so that one can create very nice images with little effort.
- The basic functionality of the rendering pipeline though is just that it can fill shapes and draw images. This simplicity combined with a bit of flexibility is what gives it its power.

# Geometry



- Graphics2D uses a powerful Shape interface to represent almost everything that gets drawn. The shape is defined by a path, the edges of which can be straight lines or polynomial curves.
- Many standard shapes are defined in `java.awt.geom`. You can also combine shapes to form areas of union, intersection, etc.

# Painting and Stroking



- A Shape defines some geometry, but not how it is drawn. How a shape is filled is defined by the painting settings. How it's outline is drawn is given by the stroking settings.
- Stroking actually is reduced to painting.
- Paint objects can be solid, gradients, or textures. Strokes can also have thickness and end styles set.

# Rendering



- Once we have shapes defined and we know what style to draw them with we have to render this. That requires transforming, compositing, clipping, and giving hints.
- Compositing decides how things are drawn on top of one another. Clipping can be done to any shape. Hints are beyond what we can talk about.

# Affine Transformations



- Drawing to a Java2D surface is done in a “user space” instead of in device space. We can specify transformations that are applied to all things that are drawn. These include translating, shearing, scaling, and rotating.
- This type of behavior is very standard in computer graphics and is typically done with matrix multiplications.

# What I've Left Out



- I haven't covered anything dealing with text or images in Java2D. These go through much of the same rendering pipeline. Text is actually drawn as shapes. There are also a number of image processing tools included in the API. This gives you a lot of power, but might take a while to learn how to use properly.



# Code



- Let's look again at the code that we have that uses Graphics2D. In addition, we will continue to write the code for transformations and think of what else we could add in to use the power of Java2D.

# Minute Essay



- How might you use some of what we talked about today in your game? In particular, how could you use it for your game status panel?
- Assignment #5 is due today. On Thursday we will venture into the realms of recursion again.