

Graphics in Java

In this assignment, you'll learn how to use Java's Graphics class, which lets you draw custom images on the screen.

Start the assignment in-class and finish the rest for homework.

Refer to Eck Chapter 6 for more information about drawing and the Graphics class in Java.

Step ONE: Make a new NetBeans project:

Launch NetBeans and create a new project (called GraphicsApp) in the usual way, with default main class graphicsapp.Main. Select your part of the class repository as the Project Location.

Step TWO: Use the Graphics class to create a custom JPanel that lets you draw on it

1. Modify the Main class (edit Main.java) – make it into a JPanel.
2. Modify the main method so that it creates an instance of a Main JPanel and shows it. Set the background color of the JPanel to red or some other color so that you can see it after adding it to the new JFrame.
3. Override the paintComponent method. In the over-ridden method, invoke the parent class's paintComponent method, which, in the case of a JPanel, will just fill in the background color. Next, try out some of the Graphics methods, starting with fillRect, which draws a filled Rectangle. (See Eck chapter 6.3.4 for some description of these methods.) Here is an example. Note that that Graphics object already has a Color, and to get the fillRect method to draw a blue rectangle, we first invoke the Graphics' object's setColor method and then set it back to the old Color when we're done:

```
public void paintComponent(Graphics g) {
    super.paintComponent(g);
    Color oldcolor = g.getColor();
    g.setColor(Color.blue);
    int width = getWidth();
    int height = getHeight();
    int x = 10;
    int y = 10;
    g.fillRect(x,y,width-2*x,height-2*y);
    g.setColor(oldcolor);
}
```

4. Now, let's implement some interactive drawing. Modify your Main class so that now it implements the MouseListener interface. (Recall that NetBeans will fill in all the abstract methods for you, but you should comment the throw UnsupportedOperationException statements.

5. Edit the mousePressed method so that wherever the user clicks, the Main JPanel draws a red, filled rectangle:

```
public void mousePressed(MouseEvent arg0) {
    System.err.println("mousePressed");
    Graphics g = getGraphics();
    int x = arg0.getX();
    int y = arg0.getY();
    Color oldcolor = g.getColor();
    g.setColor(Color.red);
    g.fillRect(x, y, 50, 50);
    g.setColor(oldcolor);
}
```

6. However, it is not enough to implement the MouseListener methods. The Main JPanel has to listen for those events. It may seem a bit circular, but in order for it to respond, it has to listen to itself. So, in the main method, add the Main JPanel to itself as a listener.

Step THREE: Turn it in using subversion.

As before, just check in some of the directories NetBeans creates when you make a new project. Review the instructions for checking in NetBeans projects from the *NetBeans and Java* assignment?

From: *Netbeans and Java* assignment

Checking in the NetBeans project files - just the ones you need!

- Change directory into your top level homework3 directory and run non-recursive svn add command as follows. This ensures that only the top level directory will get added to the repository.

```
svn add -N HelloWorldApp
```

- Change into the HelloWorldApp directory and use `svn add` to add the `src` directory, which contains all the `.java` source (program) files.
- Non-recursively add the `nbproject` directory, add the `build.xml` project file and `manifest.mf` files, and recursively add the `src` directory and its contents, like so:

```
svn add -N nbproject
svn add src
svn add build.xml
svn add manifest.mf
```

- Change into the `nbproject` directory and add the following four files, like so:

```
svn add project.xml
svn add genfiles.properties
svn add build-impl.xml
svn add project.properties
```

Don't add `nbproject/private`, `build`, or `dist` directories to the repository.

And finally, don't forget to check in all your changes in the usual way.

If you do all this correctly, then you will be able to check out and develop the HelloWorldApp project onto different computers. The key here is to check in the files that are *not* system-specific. This will make it possible for the instructor to run (and grade!) your projects and will also make it possible for you to develop your projects on different computers, using subversion to track your files.

Drawing and coordinates:

Each component (JPanels included) is drawn according to a grid, where you can address each square (pixel) of the grid using x and y coordinates:

