

# Java 2D Drawing

Note: to use any of the shape classes (e.g., `Rectangle2D.Double`, as in problem 4), you have to add “`import java.awt.geom.*;`” to whatever import statements you would otherwise be using.

1. Change your first tic-tac-toe applet (from the *first* applet exercises) to draw 10-pixel-thick lines instead of 1-pixel-thick lines. Or, grab my version from the applet-exercises project. Since we are not (yet) using Swing, you will not change `paint` to `paintComponent`, but you will still need to cast `Graphics` to `Graphics2D`. So, your `paint` method will look like this:

```
public void paint(Graphics g) {
    Graphics2D g2d = (Graphics2D)g;
    g2d.doSomethingCool(...);
    ...
}
```

2. Change the applet to use dashed lines.
3. Create a `JLabel` object (`new JLabel("some text")`) and drop it into a `Frame` that is using `FlowLayout`. Use several different labels with different font sizes and system-specific fonts. Note that to use `JLabel`, you need “`import javax.swing.*`” at the top. The `Font` constructor is used in the notes, but here is a summary:

```
Font font = new Font("some font name", Font.PLAIN, someSize);
someJLabel.setFont(font);
```

You can figure out the names and appearance of the fonts installed on your PC by bringing up PowerPoint, selecting some text, then looking at the font dropdown box at the top.

4. Make an application that has a red background color. Have your `paint` method draw 20 blue rectangles, each of which have a random transparency.
5. Make a small application that has text in various fonts drawn at various angles.
6. Modify your application from the previous problem so that you can select whether or not anti-aliasing is used. Try it both ways and see if the difference is noticeable (it should be!).