Graphics can be simple or complex, but they are just data like a text document or sound.

Java is pretty good at graphics, especially for the web and small devices like phones.
Java Graphics

- A fantasy massively multiplayer online role-playing game, 2001
  http://www.runescape.com/
- Minecraft sandbox game, 2009 https://minecraft.net/
- Instagram, 2011 - online mobile photo-sharing service
- CG Art http://www.cgartworld.com/
Java Graphics

- Java can write applications or applets.
Java can write applications or applets.

Java applet is a program that is embedded into an HTML document, and executed using a Web browser.
Java Graphics

- Java can write applications or applets.
- **Java applet** is a program that is embedded into an HTML document, and executed using a Web browser.
- **Java application** is a stand-alone program that is executed using Java interpreter.
Java can write applications or applets.

Java applet is a program that is embedded into an HTML document, and executed using a Web browser.

Java application is a stand-alone program that is executed using Java interpreter.

Java can make graphics in either one, and has two libraries to do it with:

- Swing (the newer kind) or AWT (Abstract Windowing Toolkit, the older kind).
Java - why two?

- Why is there a Swing and an AWT?

AWT was the original graphics library for Java. However, all programs looked like their “host”. That is, when they ran on a Mac, they looked like Mac programs, and when they ran on Windows, they looked like Windows programs.

Programmers wanted to force the “look and feel”, so they built Swing on top of AWT. Yes, Swing includes all of AWT.

AWT is still seen as better for applets, but that’s fading due to AJAX, another web programming technique.
Why is there a Swing and an AWT?

AWT was the original graphics library for Java. However, all programs looked like their “host”. That is, when they ran on a Mac, they looked like Mac programs, and when they ran on Windows, they looked like Windows programs.

Programmers wanted to force the “look and feel”, so they built Swing on top of AWT. Yes, Swing includes all of AWT.

AWT is still seen as better for applets, but that’s fading due to AJAX, another web programming technique.
To start, here’s a basic applet that demonstrates Java graphics using Swing

```java
import javax.swing.JApplet;
import java.awt.*;

public class BasicGraphics extends javax.swing.JApplet {
    public void paint(Graphics g) {
        /* Draw the square. */
        g.setColor(Color.red);
        g.fillRect(10, 20, 40, 40);
    } // end paint()
} // end class BasicGraphics
```
Java Graphics API

- The first lines are import statements, to load the AWT library and swing’s JApplet class.
The first lines are import statements, to load the AWT library and swing’s JApplet class.

Our class is called BasicGraphics, and it extends JApplet to inherit JApplet properties.
Inheritance Idea

- When you want to create a new class and there is already a class that includes some of the code that you want, you can derive your new class from the existing class.
- In doing this, you can reuse the variables and methods of the existing class without having to write them yourself.
Inside the applet, we have just one method: `paint()`.

It has one parameter, called the “abstract Graphics object”, and we call it “g”.

You need to tell `paint()` what to paint on!
Inside the applet, we have just one method: `paint()`.

It has one parameter, called the “abstract Graphics object”, and we call it “g”.

You need to tell `paint()` what to paint on!
paint() has two methods inside of it: setColor and fillRect

g.setColor(Color.red); is the command to color whatever graphic thing we have red. The computer still doesn't know what Graphic thing g is going to be!
Java Graphics API

- `paint()` has two methods inside of it: `setColor` and `fillRect`.

- `g.setColor(Color.red);` is the command to color whatever graphic thing we have red. The computer still doesn’t know what Graphic thing `g` is going to be!

- `g.fillRect(10, 20, 40, 40)` tells the applet to make `g` into a `fillRect`, which is a “filled rectangle”.
  10 is the starting x-position in the applet,
  20 is the starting y-position in the applet
  40 is the width and the other 40 is the height.
  Remember what color will fill it? Red!
How do you remember all of this stuff?

You don’t!

All of this is listed in the Java API.

Want to see a simple edit? Change `g.fillRect` to `g.fillOval` in the previous program.

You can do whatever changes you want, just find it in the API and make sure your parameters are filled in correctly!
How do you remember all of this stuff? You don’t!
How do you remember all of this stuff? You don’t!

All of this is listed in the Java API.

Want to see a simple edit? Change `g.fillRect` to `g.fillOval` in the previous program.

You can do whatever changes you want, just find it in the API and make sure your parameters are filled in correctly!
Applets need a web page to run, but applications can run on their own.
We can run our previous program by creating an application that would call it.
public static void main(String[] args)
{
    JFrame window = new JFrame("Janyl Jumadinova ");

    // Add the drawing canvas and do necessary things to
    // make the window appear on the screen!
    window.getContentPane().add(new BasicGraphics());
    window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    window.setVisible(true);
    window.pack();
}