

Drawing pictures ... It's not art, it's fun

Basic Processing ...

Lawrence Snyder
University of Washington, Seattle

Processing ...

- It's our main programming language
- "Processing" is kind of a dumb name,* but it is a good (and fun) language
- It's a language for programming graphical and image-based computations
 - More fun than programming an operating system
 - Easier to do because we "see" what's happening
 - It's real – it's great prep for the future

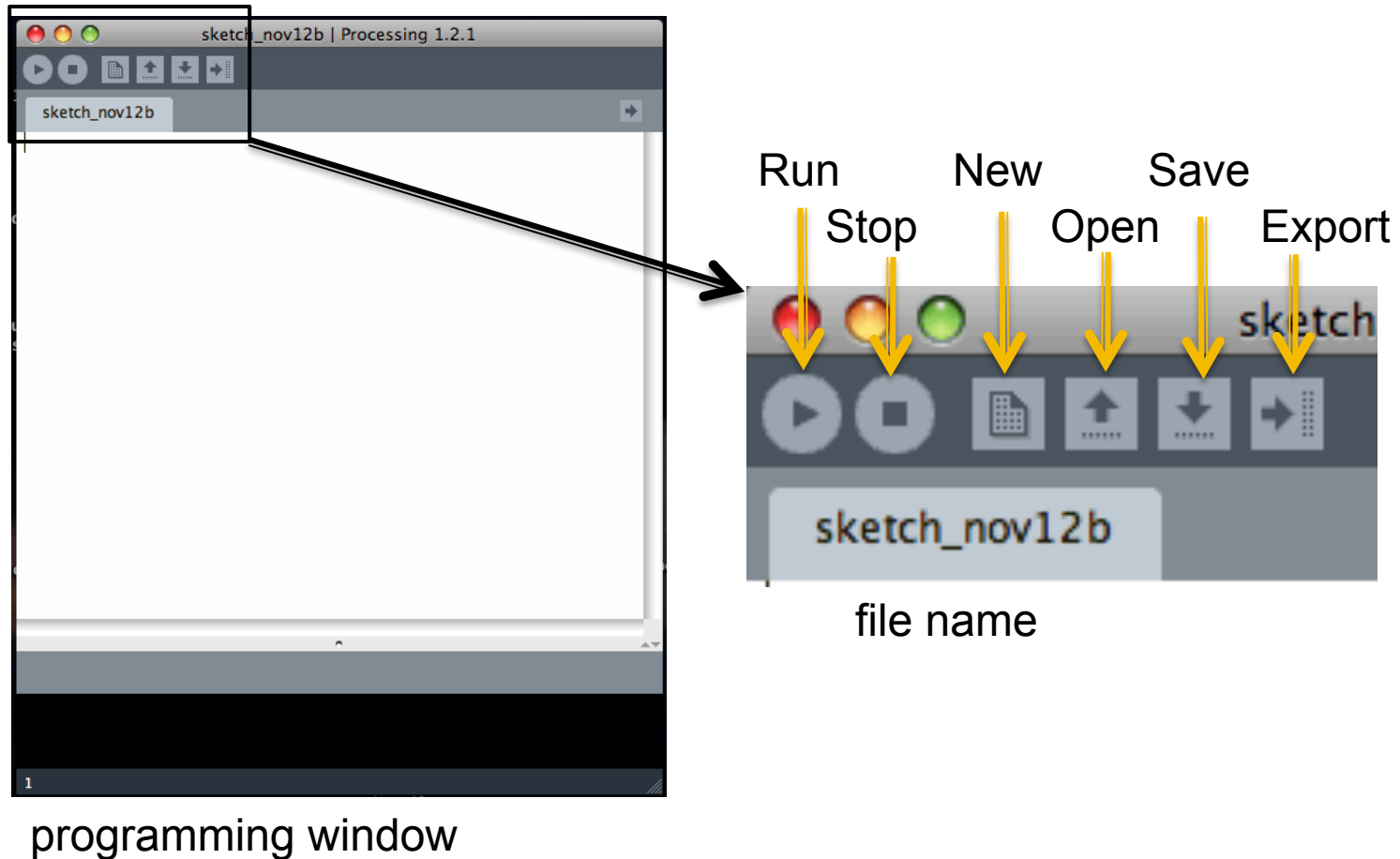
* Really really dumb, actually

Short Interrupt: Grab Processing

- If you have a personal computer that is convenient to do homework on, then grab a copy of the Processing system and put it on your machine ... improve your convenience!
- Grab it at: <http://processing.org/download/>
- You will want “Windows” or “Mac” versions
- Follow installation instructions ... it takes less than 5 minutes and then you can work on your own computer!

What You See

- When you start up the Processing system...



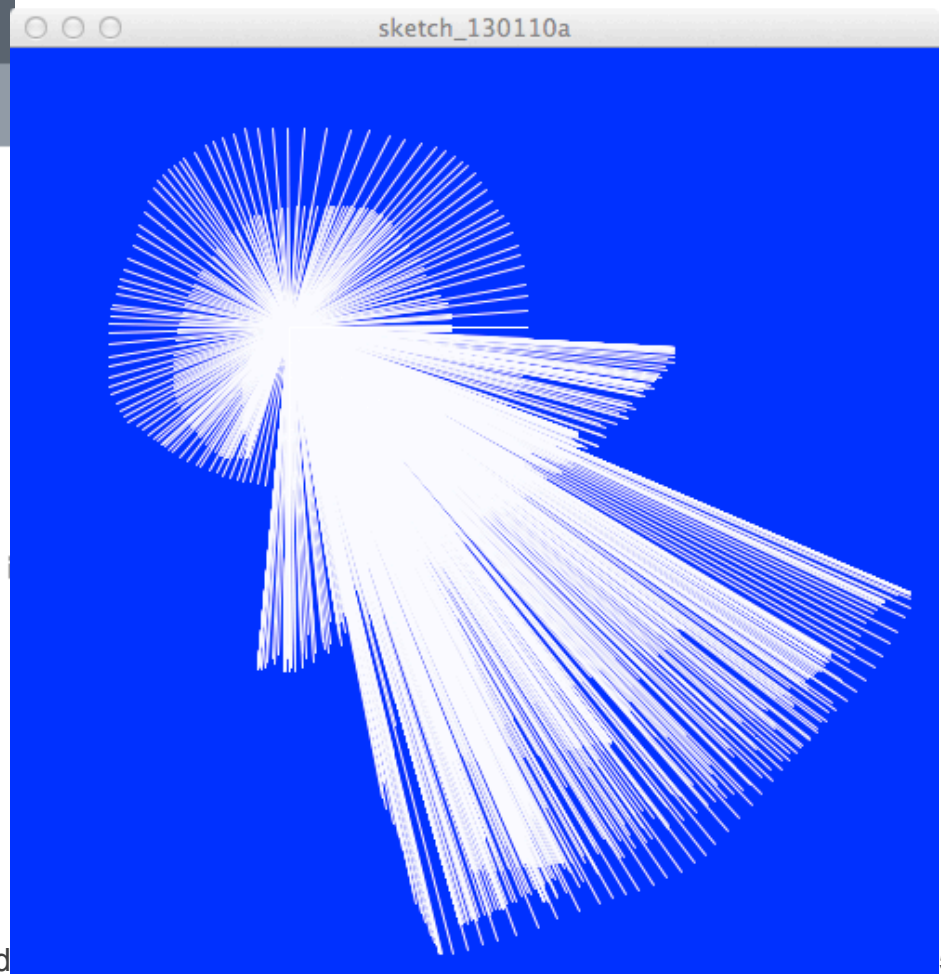
Add Some Code

- Type in instructions that you will learn shortly
Then run your program

```
sketch_130110a | Processing 2.0b
sketch_130110a §
// Draw a snow angel on blue snow

void setup() {
  size(500,500);      //define canvas size
  background(0,0,255); //define canvas color
  stroke(0,0,255);    //define line color
}

void draw() {
  line(150,150, mouseX, mouseY); //draw line from
  if (mousePressed){ //if the mouse is ever cl
    stroke(255);
  }
}
```

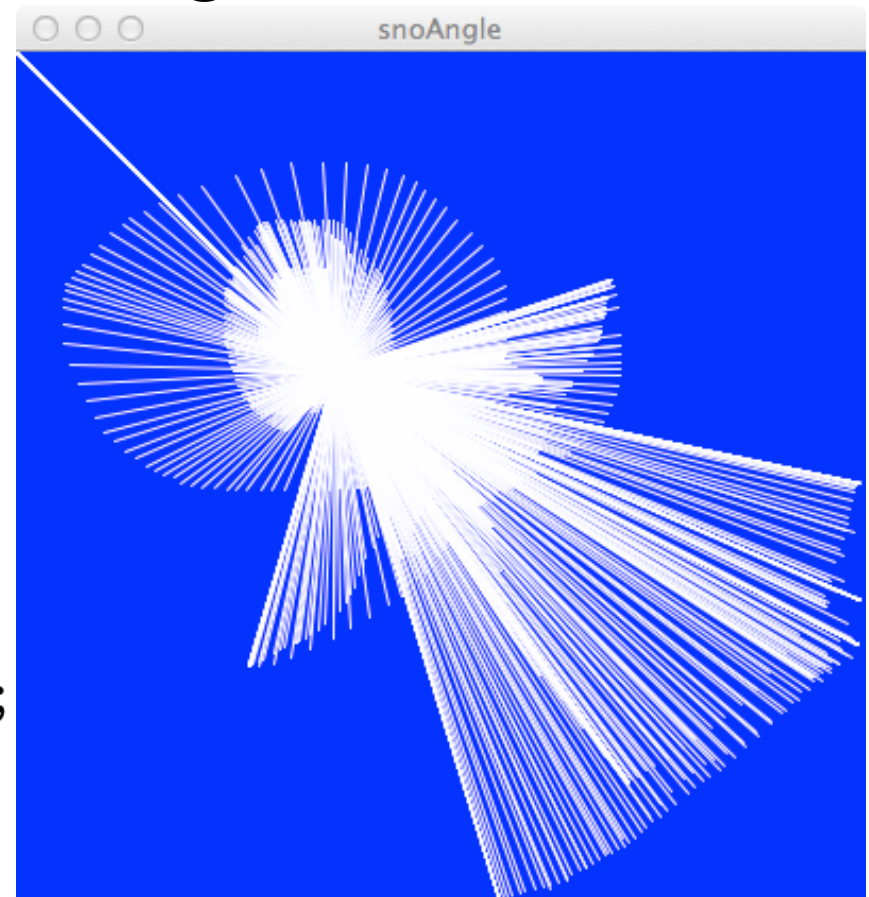


Looking At Simpler Code

- Drawing a snow angel is straightforward ...

```
void setup ( ) {  
  size(500,500);  
  stroke(255,255,255);  
  background(0,0,255);  
}  
  
void draw ( ) {  
  line(150,150,mouseX, mouseY);  
}
```

Just Do It!



Coding Is ALL Detail

■ Notice everything!



```
void setup( ) {  
    size(500,500);  
    stroke(0,0,255);  
    background(255);  
}
```

```
void draw() {  
    line(100,100, mouseX, mouseY);  
}
```

- ◆ Keywords are highlighted in blue
- ◆ Processing is case sensitive; notice!

- ◆ Two Functions, One Common Form:

```
void <name> ( ) {  
    all symbols +  
    placement matter  
}
```

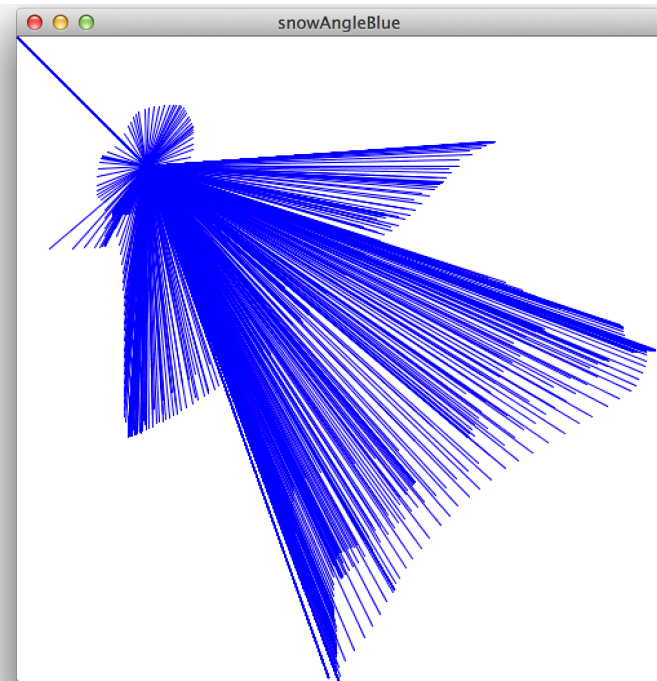
- ◆ Every statement ends with a semicolon (;)
- ◆ The software colors text it understands – helpful
- ◆ Some functions include stuff inside parentheses; these are called *arguments*
- ◆ If a function has arguments, each position has a specific meaning: *size*(<width>, <height>);
stroke(<red value>, <green value>, <blue value>);
- ◆ If your cursor is by a closing parenthesis or brace, the matching parenthesis or brace is highlighted

The Color Purple

- Colors in most Web programming are given as three values in $[0, 255]$: **RGB**, for red, green, blue
- The Color Purple, for example, is: **128,0,128**
- These positions are the intensity of the little lights that make up a pixel on the screen
 - The least intensity is 0, that is, off
 - The greatest intensity is 255, maximum brightness
 - Amazingly, the three max RGB colors make **white**
 - So, purple is $\frac{1}{2}$ intensity of **R**ed, no **G**reen, and $\frac{1}{2}$ intensity of **B**lue ... makes sense

Questions about “Angelic Huskies”

- The angel is blue on a white background specified by: `background(255, 255, 255); ...` which means?
- Stroke sets line color: `stroke(0, 0, 255);`
- Suppose it's a Husky angel on white snow:
- Stroke sets object's color: `stroke(128, 0, 128);`

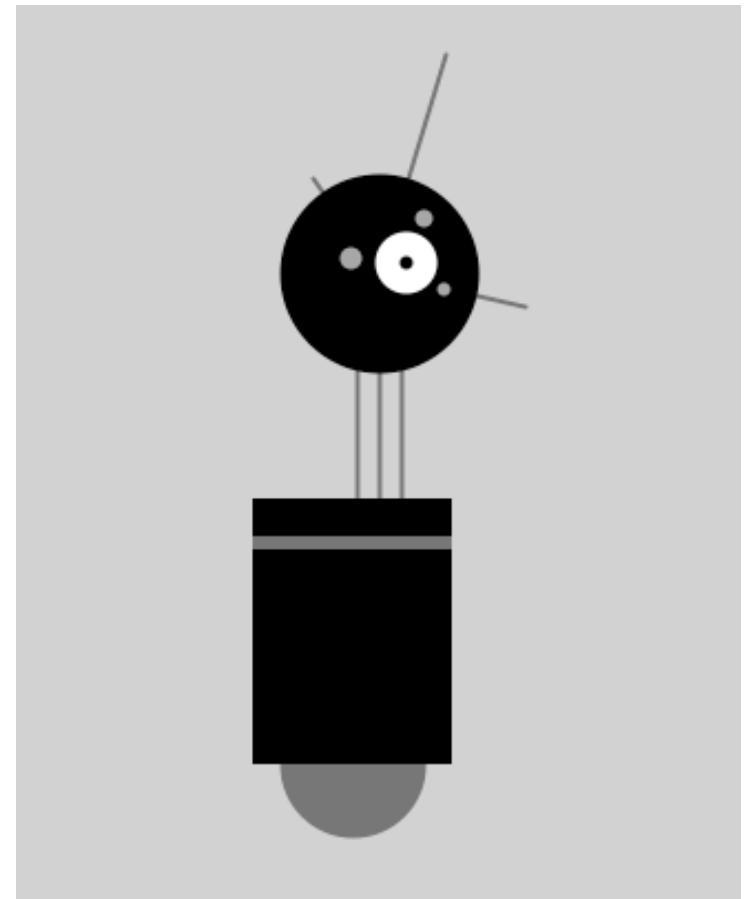


It's All The Same

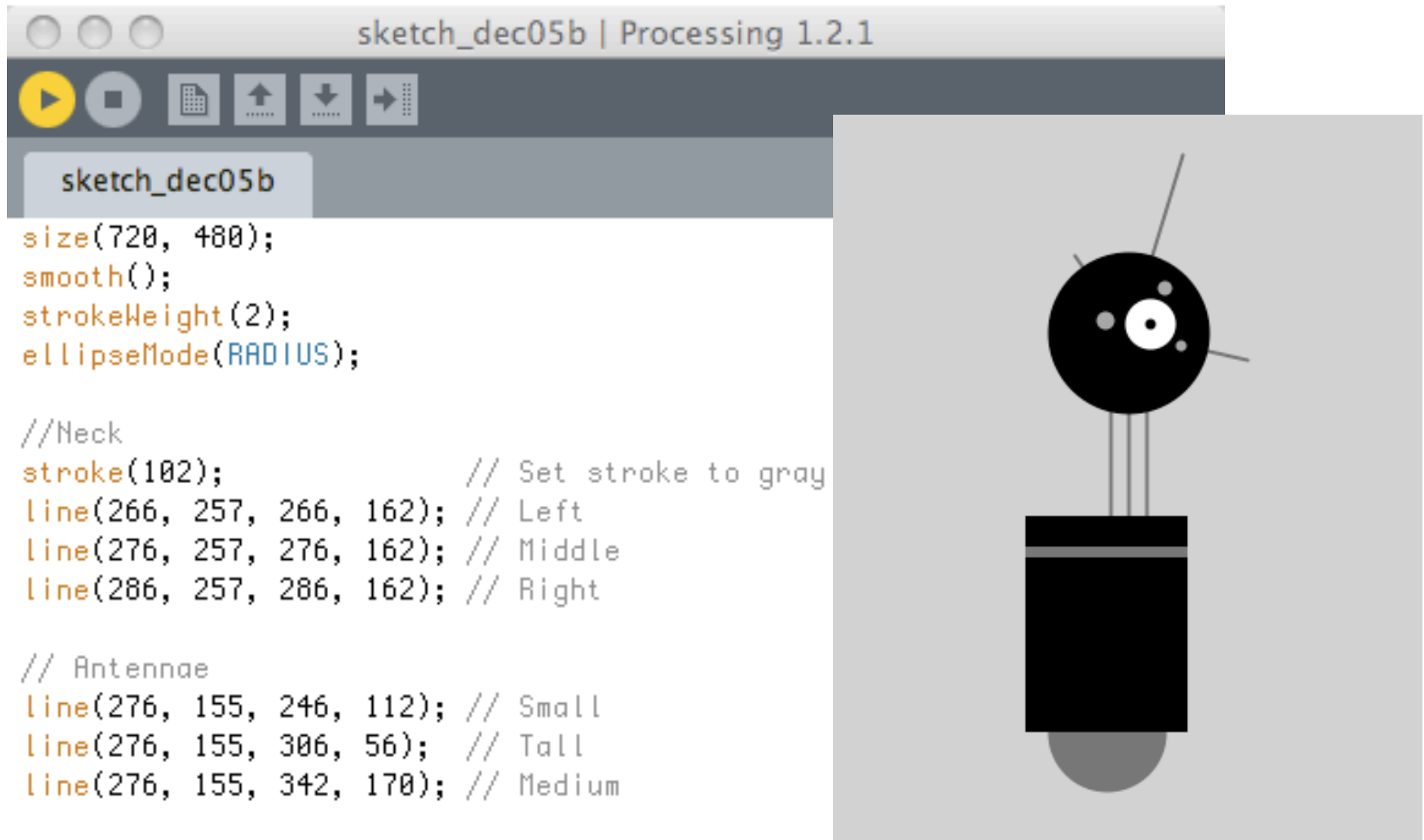
- When the values for RGB are all the same, it's some color of gray, or white, or black
- Since writing `background(255, 255, 255)` is kind of a drag, Processing allows us to give just one argument; so `background(255)` is equivalent to giving all three 255s
- What colors are these backgrounds?
 - `background(255, 0, 0);`
 - `background(64);`
 - `background(0, 0, 64);`
 - `background(255, 255, 0);`

Simple Shapes Make Robots

- Reas and Fry, in their book, show us a cute robot they programmed using simple shapes
- They give their code and we can see how they built it
- To make the point that **all code must “make sense”** – its not gibberish – lets look at it even though we don't know Processing yet



Robot Code, 1



```
size(720, 480);
smooth();
strokeWeight(2);
ellipseMode(RADIUS);

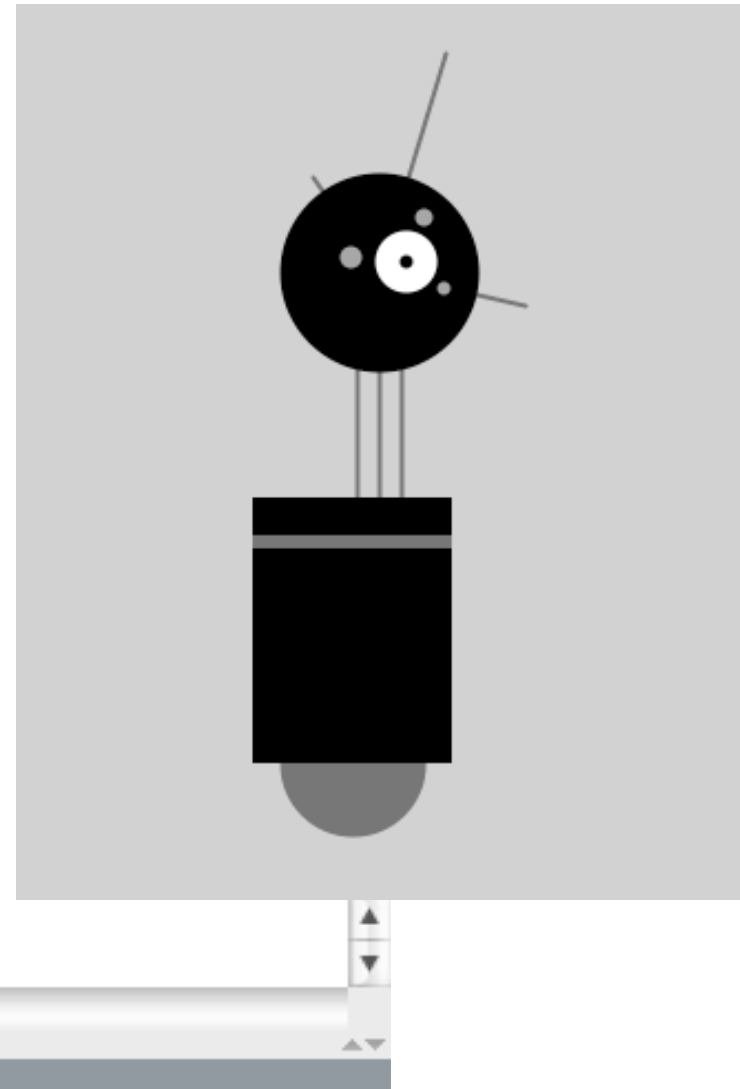
//Neck
stroke(102);           // Set stroke to gray
line(266, 257, 266, 162); // Left
line(276, 257, 276, 162); // Middle
line(286, 257, 286, 162); // Right

// Antennae
line(276, 155, 246, 112); // Small
line(276, 155, 306, 56);  // Tall
line(276, 155, 342, 170); // Medium
```

Robot Code, 2

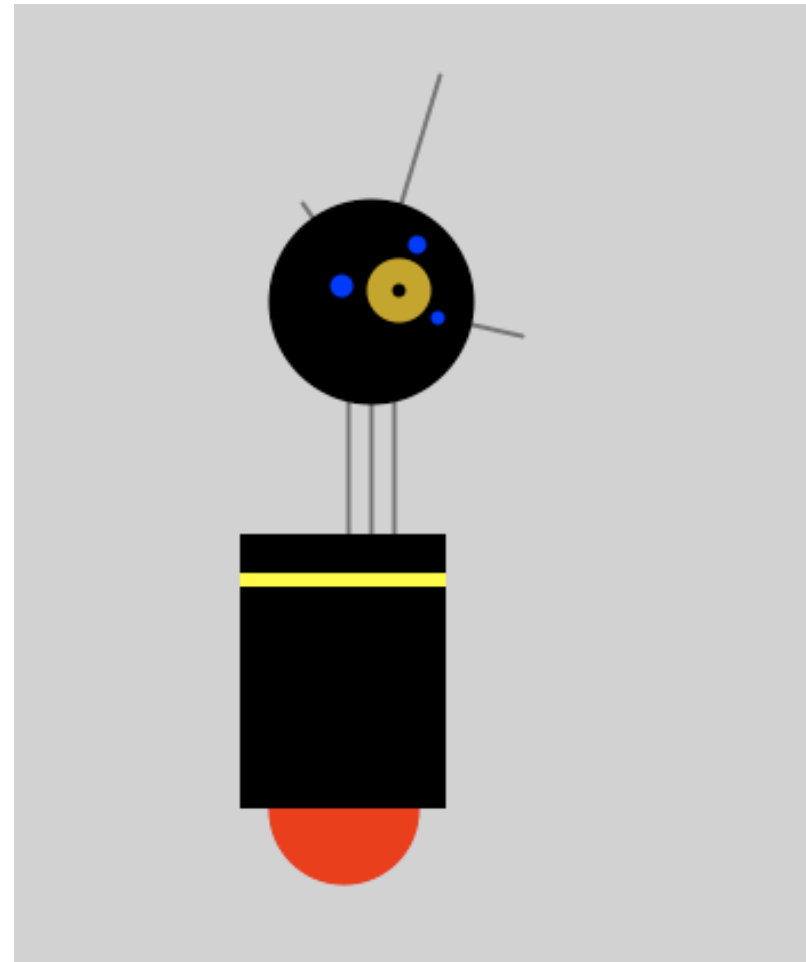
```
// Body
noStroke(); // Diabile stroke
fill(102); // Set to gray
ellipse(264, 377, 33, 33); // Antigravity Orb
fill(0); // Set to black
rect(219, 257, 90, 120); // Main body
fill(102); // Set back to gray
rect(219, 274, 90, 6); // Gray stripe

// Head
fill(0); // Set to black
ellipse(276, 155, 45, 45); // Head
fill(255); // Set to white
ellipse(288, 150, 14, 14); // Large eye
fill(0); // Set to black
ellipse(288, 150, 3, 3); // Pupil
fill(153); // Set to gray
ellipse(263, 148, 5, 5); // Small eye 1
ellipse(296, 130, 4, 4); // Small eye 2
ellipse(305, 162, 3, 3); // Small eye 3
```



Knowing Only About Color ...

- We “improve” the robot by adding some color



Just Do It!