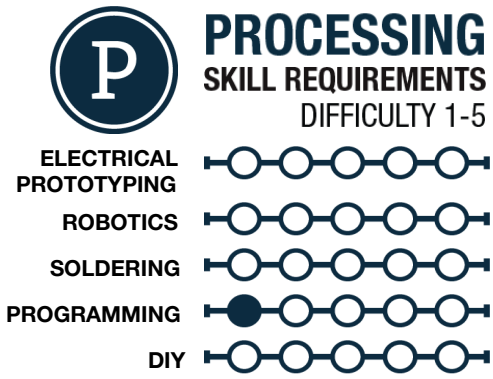
Project 0

Using computer code to make art? It's possible! Processing is a text-based coding platform for artists, designers and hobbyists to create their own visual software programs.

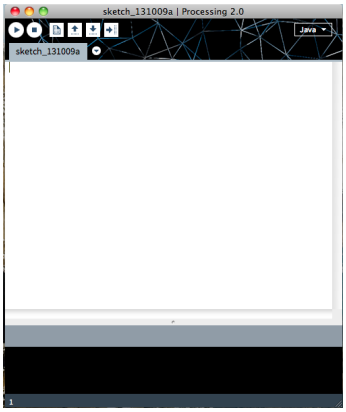


MATERIALS LIST

- Computer
- Processing

STEP 1: Opening Processing

Once you have downloaded and unzipped the Processing folder, open it and you will find the Processing logo. Double click on it and it will launch Processing!

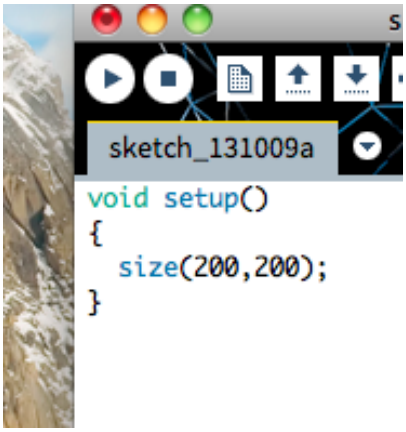


STEP 2: The IDE

Once Processing launches you will see the IDE (Integrated Development Environment) window. It should look like this. It has a text area in the middle, a play and stop button at the top, and a terminal window at the bottom. For now you should only really care about the play/stop and the text window.

STEP 3: Setup

You can click in the text area to start typing; it works just like any other text editor. Now we are going to get started coding! The first thing we have to do is write some setup code. Setup code is like building roads for the computer to follow - if I gave you directions without any roads, you wouldn't be able to get anywhere! We create setup code by typing the following into the text editor. The `void setup()` starts the setup code, and will run whatever code is in between the curly brackets { }. In this case it is the `size()` function, which dictates the size of our program window in pixels. This window will be 200 pixels x 200 pixels. Your setup code will only run once at the beginning of your program and then the computer moves on to your `draw()` code.

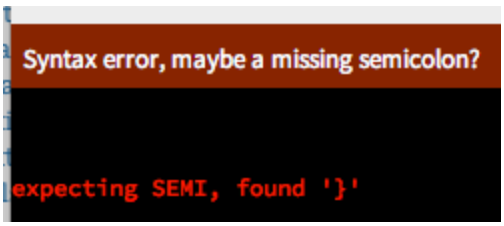


STEP 4: Syntax

Do you notice the semicolon (;) at the end of the draw function? That tells the computer that it has reached the end of an instruction and to go to the next line. If that isn't there we end up getting an error. The same holds true with the parentheses and commas, and all functions are case sensitive.

STEP 5: What Function?

A great way to find out the correct syntax and use for a function is to use the reference tool. You can find it by highlighting a function like size and right clicking on it. You can then click on "Find in Reference" from the drop-down menu. You can also click here for the reference page on the Processing website. This webpage is helpful because it gives you example code, proper syntax, and the parameters you need to give the function in order for it to work.



STEP 6: Draw

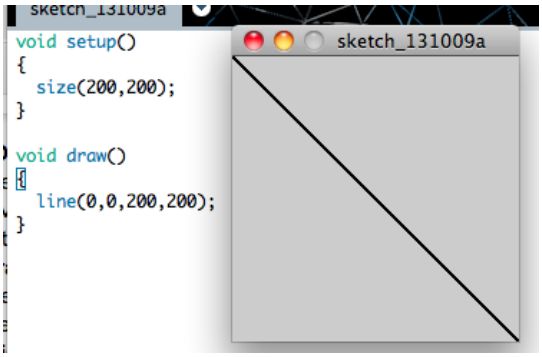
The second part of the sketch is the `draw()` code. This code is run over and over at 30 times a second! This is where we usually put all of the code we use to create something visual. First, we are going to draw a line. We use the `line()` function as seen and we add four numbers. These numbers are the x and y coordinates for the ends of the line. (0,0) is the upper left hand corner of the window and (200,200) is the lower right hand corner of the window. For a great reference on the coordinate system for Processing, click here.

```
void setup()
{
  size(200,200);
}

void draw()
{
  line(0,0,200,200);
}
```

STEP 7: Play!

Once your code window looks like this, hit the play button. What do you see? Your window should look something like below. A grey background with a line going from the upper left hand corner to the lower right hand corner. Congratulations on your first Processing Sketch!



STEP 8: Share your work!

If you would like to share your work (which you do), you can sign up at openprocessing.org. Open Processing allows you to share your code and view the drawing online, and allows others to fork or borrow your code to manipulate and change it on their own while still keeping your sketch intact for others to check out.

OpenProcessing

TAKING IT FURTHER

- Check out some examples of Processing at www.processing.org/exhibition/
- What other geometry could we draw using Processing? (Hint: Reference!)
- Draw a second line going from the upper right corner to the lower left corner.
- Change the size of your window!