CS 101
Intro to Processing

Lecture 3
Processing - What is it?

- **Processing** is a flexible software language for learning how to code within the context of the visual arts.
- In other words, it is a programming language designed for easily creating images, graphics, and animations!
- We will do lots of this throughout the semester.
Processing - What is it?

- The visual feedback is helpful when first learning to program
- Since processing is primarily used to generate graphics, it is excellent as a “first language”
Processing- What is it?

• A language for creating images, interactive graphics, animations...
  o It’s “domain” is the computer screen
  o A computer screen is a grid of light elements called *pixels*
  o Initially, drawing in Processing is like drawing on graph paper (conceptually)
    o Q: How would we draw a line, a point, anything?
    o A: Specify the coordinates
Drawing

- Do you remember drawing on graph paper?
- Let's look back at the cartesian coordinates
  - we'll experiment on the graph paper
  - *go to Elmo*
Processing - What is it?

- **Processing** is just one of many computer programming languages
- You may have heard of some of the more popular languages such as C, Java, and Python
- If you continue on in computer science, you will learn all of these languages, and perhaps more!
- For this class, we stick primarily to **Processing**
Processing - What is it?

- **Processing** is actually a **dialect** of Java
- This means the the **syntax** (the way the code looks and is structured) is very similar to Java
  - You probably don’t know java, but when you someday learn it, you’ll see the clear resemblance
  - Processing isn’t Java :)


Processing - The Textbook

- This is our textbook
- We will be following the order and structure of this book closely while learning processing
- This book has great examples and visual aids to help solidify concepts, so make sure and do the readings!
Along with the textbook, processing.org will be a great resource for you to learn processing:
- This site has great learning resources, code examples, tutorials, and more
- You can also download the processing IDE from here
The first step to get started with processing is to download and install the processing IDE (Integrated Development Environment)

- The processing IDE is the program you will use to both write processing code and to run the programs you write

Start at processing.org

- https://processing.org/download/
Processing - How does it work?

- In processing (as with most programming languages) we write a sequence of instructions, which get executed step-by-step, in order
  - Algorithms!
- In general, the instructions you give to a computer are executed from the top to the bottom
  - We will learn how to change this later!
Let’s go ahead and write our first program
To do so:
  ○ Open up the “Processing” app
  ○ Type in the following:
    \[
    \text{ellipse(50, 50, 80, 80)};
    \]
  ○ Hit the run button
Let’s go ahead and write our first program.

To do so:
- Open up the “Processing” app
- Type in the following:
  ```java
  ellipse(50, 50, 80, 80);
  ```
- Hit the run button
ellipse(50, 50, 80, 80);

- This line of code means “draw an ellipse, with the center 50 pixels over from the left and 50 pixels down from the top, with a width and height of 80 pixels”
- Try changing the numbers, and see what differences occur
The Canvas

- As mentioned, processing programs are visual
- Graphics are drawn onto the *canvas*
- The canvas is a grid of tiny pixels
  - Arranged in rows and columns
- We specify where we want things to be drawn on the processing canvas using pixel coordinates
- **BUT,** the coordinate system is oriented in a different way…
Left/Right is the X axis
Up/Down is the Y axis
X-axis positive

Y-axis positive
A particular position on the canvas is specified by an X position and a Y position (coordinates)

(0,0) is in the upper left corner!
X=0, Y=0
This particular processing program canvas is 600 pixels wide and 400 pixels tall

Set size with: size(x, y);
A particular position on the canvas is specified by its X position and Y position (coordinates).

For Example...
X=300, Y=0
X=???, Y=???
X=100, Y=100
X=???, Y=???
X=500, Y=370
ellipse(50, 50, 80, 80);

- Let’s return to this line of code
- What happens when we use different values for X, Y, width, and height?
- How does it change?
Three ellipses

- Try drawing THREE ellipses to the screen with different X, Y, width and height
- In other words:

```
ellipse(?, ?, ?, ?);
ellipse(?, ?, ?, ?);
ellipse(?, ?, ?, ?);
ellipse(?, ?, ?, ?);
```
Three ellipses

- Try drawing THREE ellipses to the screen with different X, Y, width and height
- In other words:

  ellipse(? , ?, ?, ?);
  ellipse(? , ?, ?, ?);
  ellipse(? , ?, ?, ?);

- What do you get?
• We can draw other shapes too:

```java
rect(x, y, w, h);

triangle(x1, y1, x2, y2, x3, y3);

line(x1, y1, x2, y2);

point(x, y);
```
Functions

- As a programmer, you tell the processing language what, where, and how to draw things by calling **functions**
- A **function** is a sequence of code that can be “called” or “invoked” by calling it
- In fact, we’ve already called a few functions
Functions

- When you call a function, you must give the function 0 or more arguments
  - A argument is a bit of information that you can give the function to control what it does
  - The order that you write argument in matters!
  - Each of the functions you’ve used take a few arguments
Functions

- **ellipse(x, y, w, h)** - A call to a function that draws an ellipse at the x/y coordinate and width/height provided
- **size(w, h)** - A call to a function that sets the size of the processing drawing canvas
- **rect(x, y, w, h)** - A call to a function that draws a rectangle at the x/y coordinate and width/height provided
- ... and more!
Drawing a simple canvas

Write a small processing program that creates the following canvas

Remember:
- `size(width, height)`
- `rect(x, y, w, h)`
- But where does x,y start?
Functions

(rect(x, y, width, height)
Drawing a simple canvas

Write a small processing program that creates the following canvas

Remember:

- `size(width, height)`
- `rect(x, y, w, h);`
- We’ll use a 600 x 400 canvas
Drawing a simple canvas

Solution:
size(600,400);
rect(40,0,520,40);
Drawing a simple canvas

Write a small processing program that creates the following canvas

Add a rectangle on the bottom

Remember:
- `size(width, height)`
- `rect(x, y, w, h);`
Functions

triangle(x1, y1, x2, y2, x3, y3)
Functions

ellipse(x, y, width, height)
Drawing a simple canvas

Write a small processing program that creates the following canvas

Remember:
- `size(width, height)`
- `ellipse(x, y, w, h);`
- `rect(x, y, w, h);`
Drawing a Snowman

• Goal: Draw an (ugly) snowman like the one to the left
• How can this be done, using what we know about processing so far?
Drawing a Snowman

- Goal: Draw an (ugly) snowman like the one to the left
- How can this be done, using what we know about processing so far?
- What can we add to it to make it look better?
Processing - Materials

- Required Materials
  - GSWP - Preface, Ch 1, Ch 2 (~20 pages total)
  - Hello Processing (tutorial) (1 hour)
  - You Should Learn to Code (TED video) (10 mins)