CART 253
Creative Computation 1

Email: lwilkins@concordia.ca
Office Hours: Tuesday 12-1
Course Github: https://github.com/LeeRobot/CART253-F-22

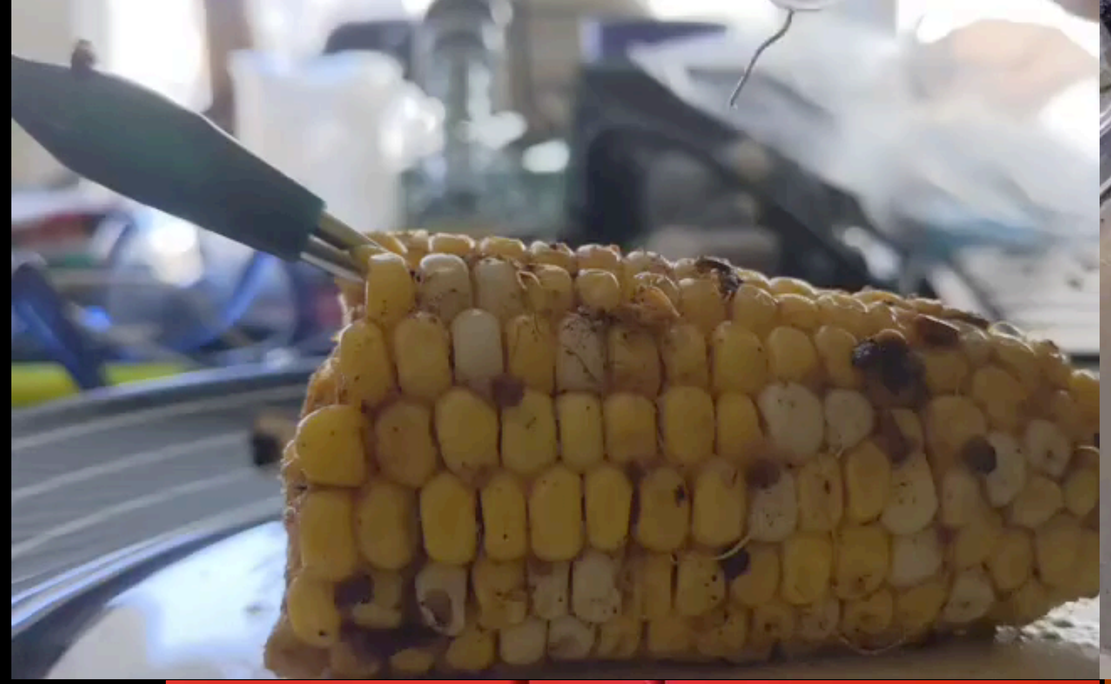
Hello!

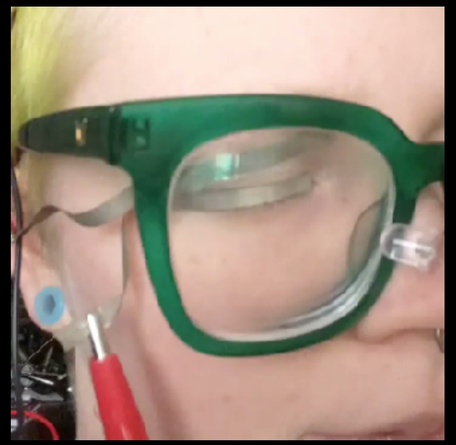
- Whats your name? Pronouns?
- What year / program are you in ?
- How do you feel about programming?
- What are you excited about this semester? (Doesn't have to be this class!)
- What is your favourite breakfast food?

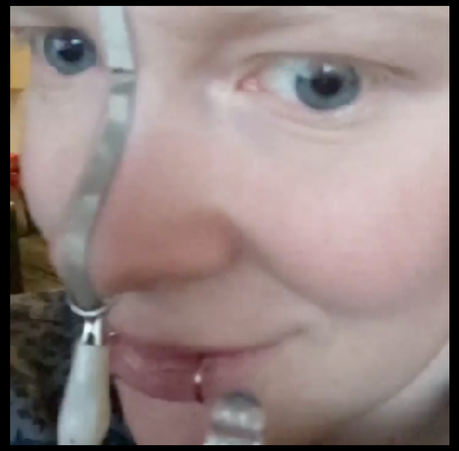
I teach at places like Concordia University, Toronto Metropolitan University, OCAD University and also School of Machines, Magic and Make Believe, and Inter/Access.

I write for Make Magazine, The Prepared & Hackaday

I run events like Dinacon, Stupid Hacks, Biohacking Village, and Open Hardware Summit






Digital
Naturalism
Conference


RE:Familiar


Android
Apparatus


Stupid Shit Nobody Needs
And Terrible Ideas Hackathon
Toronto
2020


Earth
Resist

What is creative computation 1? In this class we will learn how to use **Javascript** to make digital art! We will start with learning to control what we see on the screen, then move to interaction, and eventually interacting with the environment and each other using a web browser.

Projects & Critique:
All submissions are due on Moodle as a zipped file at midnight of the due date. You should bring your work to class to present and critique, and can make changes between your critique and final submission. See project description for individual requirements.

Exercises:
You will have in-class time to finish these basic exercises. There are 4, each worth 5% and should be zipped and uploaded to Moodle by midnight of the class.

Participation:
Participation is based on attendance, asking questions, helping peers, participating in critique, emailing the prof or TA, engaging actively with the class material.

Submissions

Projects

Portraits: 10%
Create a portrait using P5.js and basic shape and color functions

Screens in space 20% (can be done in pairs)
Create a site-specific installation that is interactive or evolving. Take a video of it, or bring us to it for critique.

Digital Rube Goldberg 40%
Pass a variable using MQTT to your classmates to create a chain of interactions

In class exercises 20% (5% x 4)

Participation 10%

Week 1: Hello World - Lets make shapes!
Week 2: Variables
Week 3: Portraits due, group critique.
Week 4: Conditionals (**In class exercise 1 due**).
Week 5: Loops (**In class exercise 2 due**).
Week 6: Arrays.
Week 7: Functions (**In class exercise 3 due**).
Week 8: Screens In Space due, group critique.
Week 9: MQTT (**In class exercise 4 due**).
Week 10: Using sound in P5.js.
Week 11: Using the camera in P5.js.
Week 12: Studio Session.
Week 13: Rube Goldberg project due, group critique.

Course Outline

Office Hours

Lee’s office hours:
Tuesdays 11-12
Via email lwilkins@concordia.ca

TA (will post to Moodle)

If you need help, ask. If you need extensions, other formats, assistance in anyway, you can contact the Accommodations office. If you need help doing that, please ask me. I can’t help if I don’t know what you need!

Accommodation

All course info, examples and everything is on GitHub
https://github.com/LeeRobot/CART253-F-22/tree/main

- Join the discord for p5.js
- Review the reference for code help
- Follow this getting started tutorial
- Review P5.js and Atom tutorial
- Check out the p5.js forums for help
- Nature Of Code examples

Resources

Loving code isn't for everyone, but you can learn a bit and **have fun** even if it isn't your passion!

Its not as scary as it seems, you need to approach it with an **opened mind**. We are here to learn and **ask questions**.

Knowing just a bit can help you **collaborate better** with others, and **imagine different ideas**. Even if you never code again after this class, you'll understand another facet of art!

Keep calm! Take a walk! **Its okay to be frustrated**, a break will help.

On learning to code...

We are all here to become better artists, designers, and/or programmers, and to help each other achieve that goal! Therefor its important we take time to reflect on our own work and our peers, which is something you'll rarely have an opportunity to do after you graduate. Approach this class with a positive attitude and understanding that we are here to help.

- Be constructive
- Be honest
- Be supportive

We don't grow by not being critical.

Code & Critique

Its also important to understand that there is a technical element to this class beyond conceptual to build what you're imagining. Why isn't your concept being achieved, is it a technical hurtle we can help with? Be honest about your intentions and accidents, we can help!

Code & Critique

Inspiration

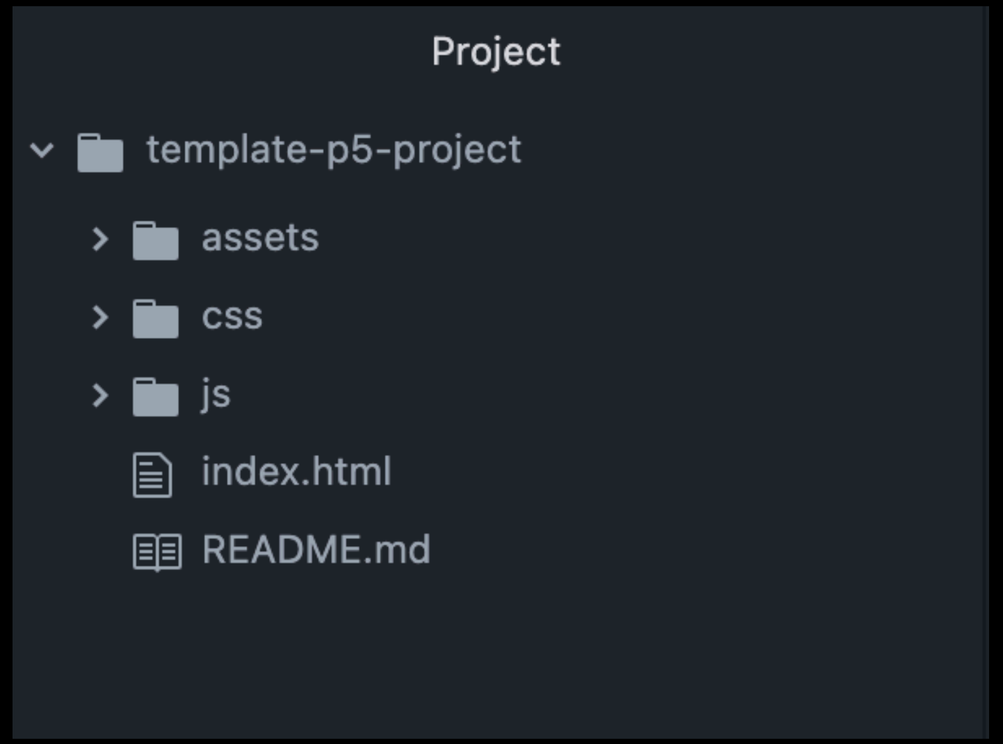
- Libbs Elliot makes quilts using generative art
- Zach Lieberman's Land Lines lets you draw any line and finds a map on Google Earth that uses that shape
- Open Processing is a place people upload their creative code projects

Getting Started

- All course information is on Moodle and here (projects, exercise, inspiration, lectures, outlines)
- Download Atom here (for writing code)
- Download the template here (for setting up your project)

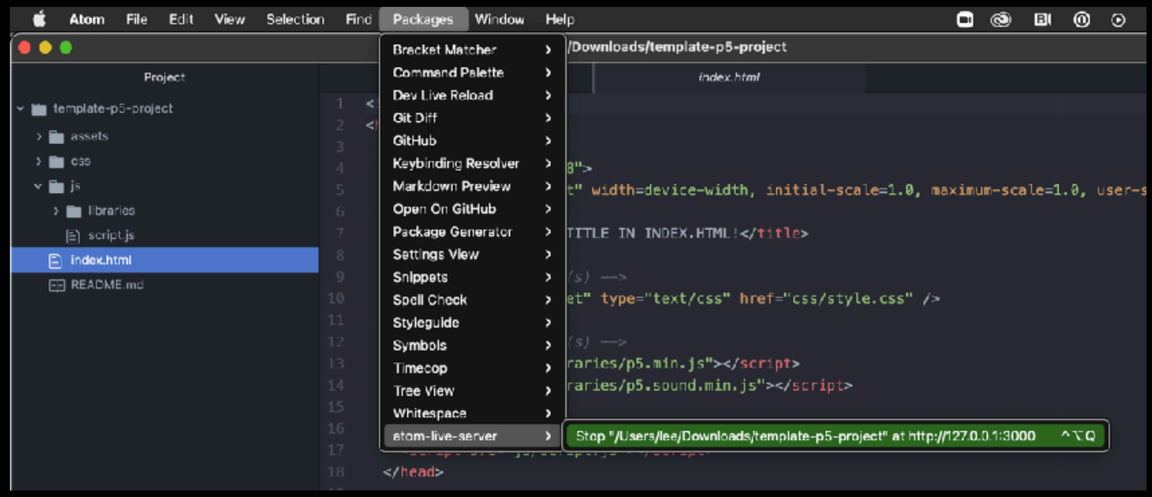
Open the template

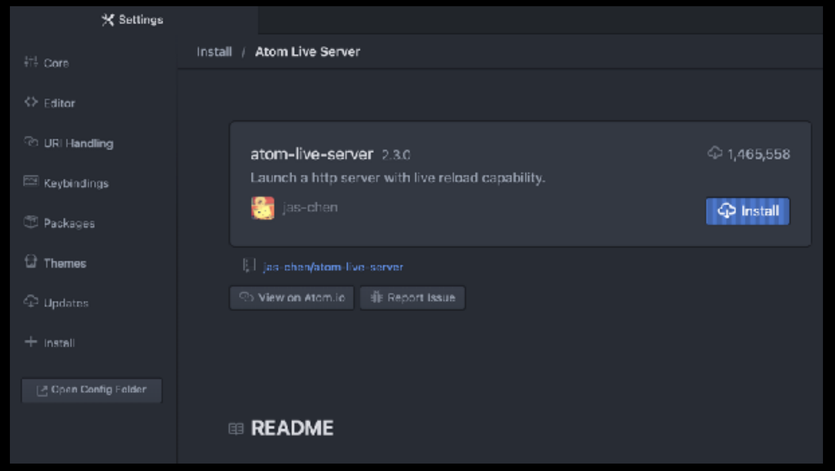
- Unzip the template
- **File > Open.** open the whole folder, not the individual files, so that the whole folder is visible in the sidebar
- Full tutorial here



Use Atom Live Server

- To run code, you'll need to either upload it to a web server, or use a local server. Atom has one already called Atom-Live-Server.
- You can run your code by clicking Packages > atom-live-server > Run
- You may be missing the package, you can download it by clicking Atom > Preferences > Install and searching Atom-Live-Server, then you'll be able to find it in the packages menu.





Run the code

- Your code will open in a browser. When you save your code, it will reload automatically. You can stop the server by going to Package > atom-live-server again

Files in the template

- Readme: This file contains meta information about the project like name, project title, how to use/install it etc.
- index.html: This file is the “shell” that holds everything, it renders all other attached files and is the scaffolding of a website. It automatically loads first.
- Js > script.js: This is where your code lives! For the most part, we will be editing this file.
- Js > libraries: Here you will find the library p5.js
- CSS: This is used to change the appearance of HTML, we’ll talk more about it later.
- Assets: Images, other scripts or things you might need.

Use the reference

https://p5js.org/reference/ Always have this open when you code!

Script.js

```
// This is a comment, the program will ignore it! Because of the slashes

function preload() {

// Anything you put inside the curly brackets is part of the function
// The preload function happens before the code is loaded.

}

function setup() {

// This is part of the setup function, anything in here happens when your code first loads. Once only.

}

function draw() {

// The draw function happens again and again. We'll explore it later.

}
```

Setup Function

Keep an eye on the **curly brackets**, they should be at the beginning and end of each function.

Comments can be added at the end of a line or on a new line. **Use them to describe what you're doing.** It might seem obvious now, but as we make more complex code it will help.

```
function setup() {

  createCanvas(500, 500); // Create the canvas

}
```

This is the **name** of the function. It often describes what it does. You can look it up in the reference. **This one creates the canvas.**

This is called a **parameter**, it is a piece of information the function needs to work. **In this case its the size of the canvas.**

There are some forms of punctuation that are required, such as **brackets around parameters, commas separating parameters, and semicolons at the end of lines.**

Find a function in the reference

When you encounter a new function, or are looking for a function, first find it in the reference! Here is the background function. You'll find examples of code, syntax (how it it goes together), and parameters, which describes what values you can give it.

So there are a few ways to use it, here are some easy ones.

```
background(grey);

background(red, green, blue);

background(red, green, blue, alpha);
```

Here, values are between 0 (off) and 255 (max)

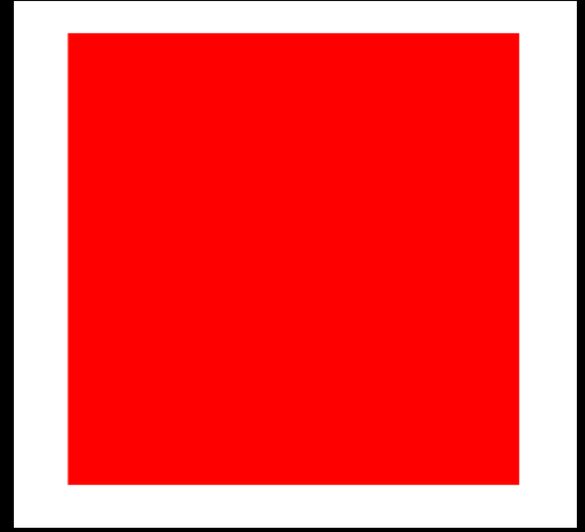
RGB values

There are multiple ways to choose a color in p5, the most common way is to use R G B values. They go from 0 to 255, and you assign how much red, green, or blue you want in your color. Example:

Bright green: (0, 255, 0) // Maximum green
Teal: (0, 150, 150) // Some green, some blue.

There is also options like using hex codes, color names, or HSV (hue saturation vibrance) to define your colours.

Setup Function



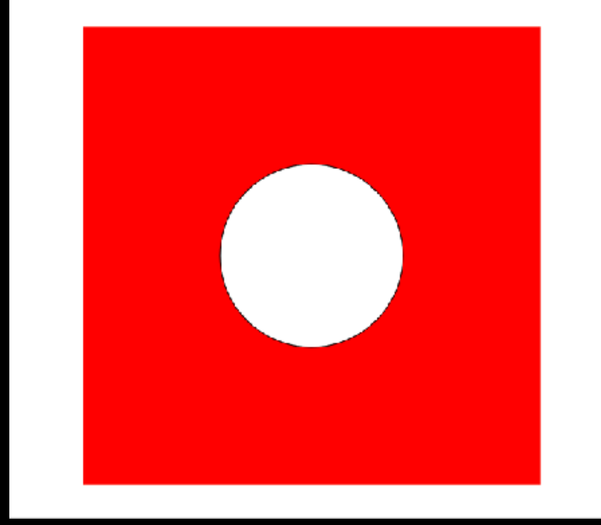
```
function setup() {

  createCanvas(500, 500); // Create the canvas
  background(255, 0, 0); // make the background red

}
```

Save, and then you can navigate back to your browser and see the 500 pixel red canvas, ready for our art!

Draw a circle



Lets try a shape function like ellipse
Syntax:
ellipse(x, y, width, height);
Parameters: x coordinate, y coordinate, diameter of circle

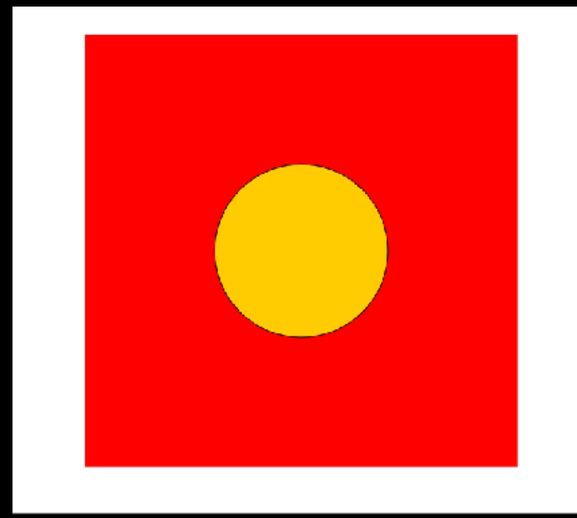
```
function setup() {

  createCanvas(500, 500); // Create the canvas
  background(255, 0, 0); // make the background red
  ellipse(250, 250, 200, 200); // Create a 200 pixel wide circle at 100 100

}
```

By default:
0, 0 is the top left.
0, 100 is 100 pixels down,
100, 100 is 100 pixels left and 100 pixels down.

Color the circle



Lets add color using fill();

```
function setup() {

  createCanvas(500, 500); // Create the canvas
  background(255, 0, 0); // make the background red
  fill(250, 204, 0); // make the background yellowish
  ellipse(250, 250, 100); // Create a 100 pixel wide ellipse at 250/250

}
```

After you set a fill color by using the function, all shapes you draw after that will be

Remove the stroke

Lets try noStroke();

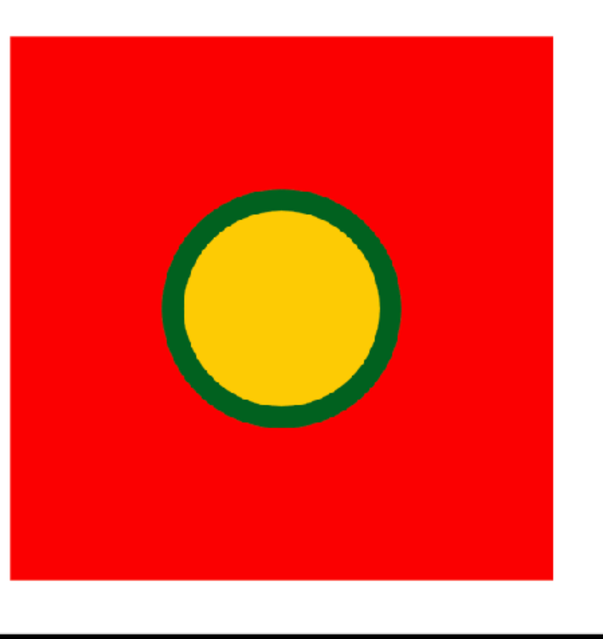
```
function setup() {

  createCanvas(500, 500); // Create the canvas
  background(255, 0, 0); // make the background red
  fill(250, 204, 0); // make the background yellowish
  noStroke(); // make the background yellowish
  ellipse(250, 250, 100); // Create a 100 pixel wide ellipse at 250/250

}
```

After you set a fill color by using the function, all shapes you draw after that will be

Change the stroke



we can use stroke() to change the color and strokeWeight()
to change the thickness

```
function setup() {

  createCanvas(500, 500); // Create the canvas
  background(255, 0, 0); // make the background red
  fill(250, 204, 0); // make the background yellowish
  stroke(0, 100, 34); // Make the stroke blue
  strokeWeight(20); // make it 20 px
  ellipse(250, 250, 100); // Create a 100 pixel wide ellipse at 250/250

}
```

After you set a fill color by using the function, all shapes you draw after that will be

Check out some of these !

arc()

ellipse()

circle()

line()

point()

quad()

rect()

square()

triangle()

bezier()

bezierDetail()

bezierPoint()

bezierTangent()

curve()

curveDetail()

curveTightness()

curvePoint()

curveTangent()

ellipseMode()

noSmooth()

rectMode()

smooth()

strokeCap()

strokeJoin()

strokeWeight()

Style & formatting

Full style guide here.

- Get rid of extra white space
- use TAB to indent functions, everything in side the function should be aligned.
- Curly brackets should be at the first line of the function, and alone at the end of the function.

```
function setup()    {
createCanvas(500,500);}
```

```
function setup() {
  createCanvas(500,500);
}
```

Resources

Topics

Instructions YouTube Playlist

Writing code

- Watch 2.1. Working with code in Atom (18:16) (and read Working with code in Atom)
- Watch 2.2. Drawing shapes (50:44) (and read Drawing shapes)
- Watch 2.3. Drawing colors (21:02) (and read Drawing colors)

Style

- Watch 2.4. Commenting your code (13:36) (and read Commenting your code)

Debugging

- Watch 2.5. The JavaScript console (16:56) (and read The JavaScript console

First Project: Portrait, due week 3

- Make a portrait of yourself, your friend, your partner, your cat, your sibling! A real person (or living creature in your vicinity), no celebrities or cartoons.
- It doesn't have to be realistic. Use shapes and explore color and form!
- Come to class ready to share on week 3, we'll talk about code and also aesthetics. Upload code to Moodle by midnight after class. Critique in class is part of your grade, so you must attend!
- Full details on Git and Moodle.

Play with color, ***shapes***, the order you draw shapes in, merging twoshapestogether, using alpha channels, try bezier curves, use **points**, experiment with stroke, try a different **style**, be abstract, try a new function, e**xplore the reference, explore the internet.**