introduction to media computing
week 01

School of creative media, mw

Course Canvas Page

https://canvas.cityu.edu.hk/courses/37822

canvas

Account
Dashboard
Courses
Calendar
Inbox
Help

202009SM1103AL02

2020/21 Semester A

- Home
- Assignments
- Discussions
- Grades
- People
- Pages
- Files
- Syllabus
- Collaborations
- Chat
- Library Resources
- Class List (AIMS)
- uReply
- Panopto Recordings
- TLQ
- Zoom
- Office 365

SM1103AL02 Introduction to Media Compt'ng

This course teaches fundamental computer coding concepts via creative exercises and small projects. Students will explore the concepts of variables, sequential programming, loops, conditionals, arrays, and functions with the programming of multimedia, such as image, audio, video, animation, and interactivity.

Section L02 Instructor - Dr. MIke WONG (smmike@cityu.edu.hk)

General Information

- Evaluation Scheme
- Readings

Weekly Meetings via ZOOM (Tuesdays 12:00pm - 2:45pm)

- Week 01

Assignments

- **TBA**

Essential Coding Topics

- Variables
- Operators
- Conditionals
- Loops
- Arrays
- Functions
- Program Flow

Resources

- p5.js Shapes Visual Reference
- Useful Snippets
- Tips & Tricks
- The p5.js website
- The p5.js reference
- The p5.js online editor
- The Atom editor website

View Course Stream
View Course Calendar
View Course Notifications

To Do

Nothing for now

Teaching Assistants

- CAI Shaoyu (shaoyu.cai@my.cityu.edu.hk)
- XIAO Chufeng (chufexiao2-c@my.cityu.edu.hk)

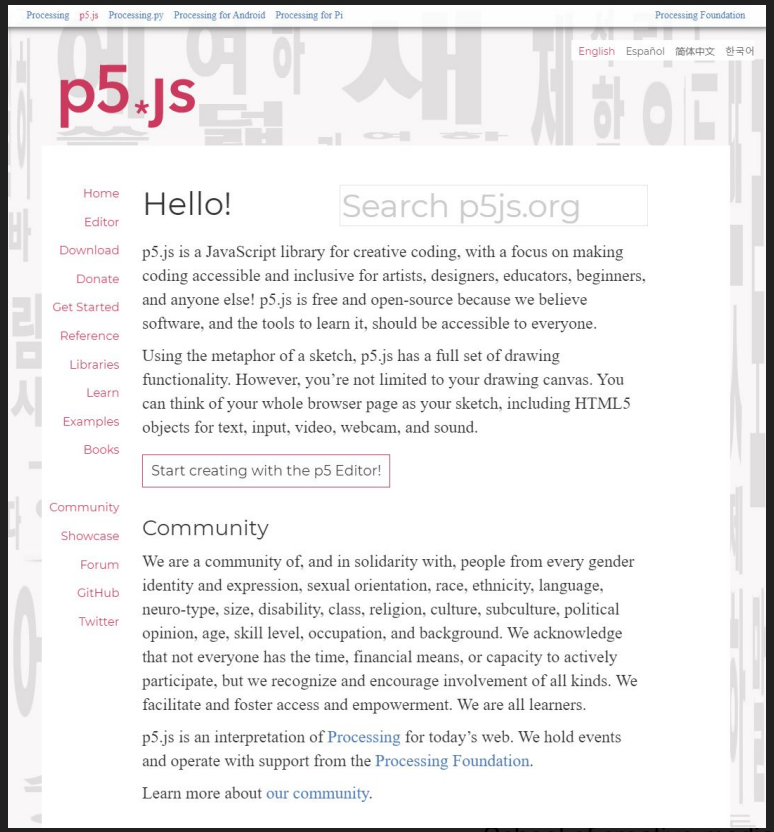
Evaluation Scheme

- Assignments (50%)
- Mid-term Quiz (20%)
- Final Exam (30%)

today's topics (week 01)

- What is p5.js ?
- p5.js online editor
- a basic p5.js sketch
- p5.js canvas coordinate system
- simple p5.js drawing

What is p5.js ?



p5.js is the JavaScript version of Processing

Processing is a popular Creative Coding environment

(demo)

HTML

Webpage in HTML

HTML / JavaScript

Webpage in HTML

JavaScript code

HTML / JavaScript / p5.js

Webpage in HTML

JavaScript code

p5.js sketch

HTML / JavaScript / p5.js

```html
<html>
<head>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.9.0/p5.js"></script>
  <script>

  function setup() {
    createCanvas(200, 200);
    rect(120, 150, 200, 300);
  }

  </script>
</head>
</html>
```

HTML / JavaScript / p5.js

open HTML tags

```
<html>
<head>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.9.0/p5.js"></script>
  <script>

  function setup() {
    createCanvas(200, 200);
    rect(120, 150, 200, 300);
  }

  </script>
</head>
</html>
```

close HTML tags

HTML / JavaScript / p5.js

```
<html>
<head>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.9.0/p5.js"></script>
  <script>

  function setup() {
    createCanvas(200, 200);
    rect(120, 150, 200, 300);
  }

  </script>
</head>
</html>
```

p5.js library from CDN

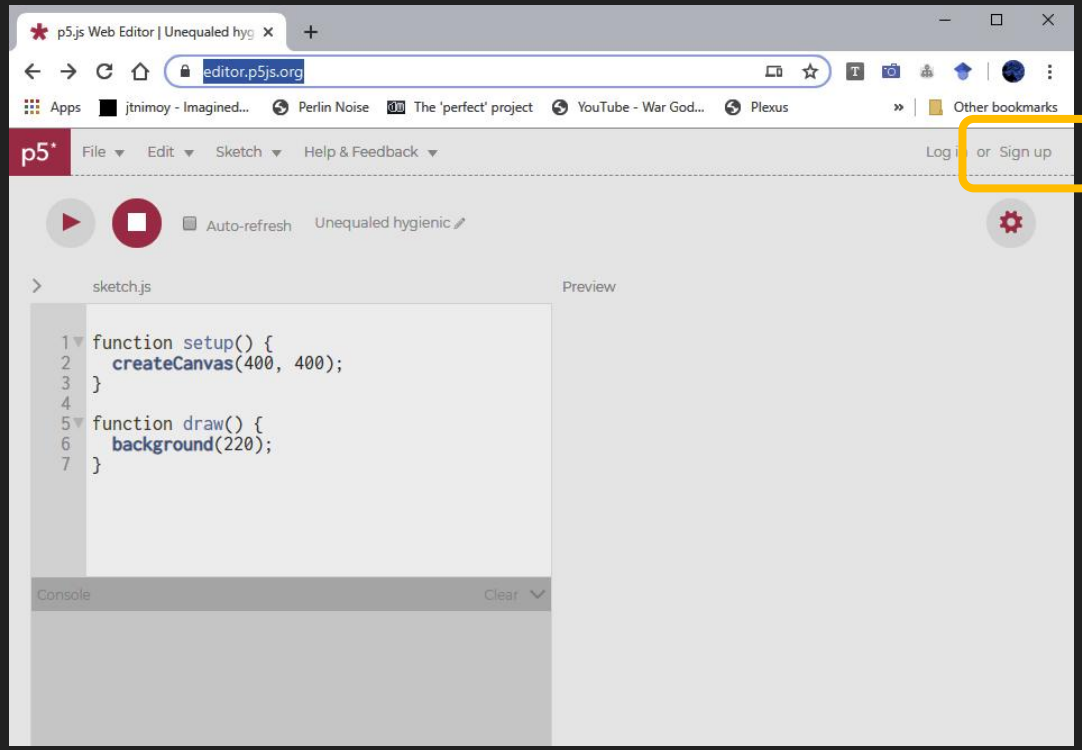
your code here

basic p5.js

```
function setup() {
  createCanvas(200, 200);
}

function draw() {
  background(random(0,255));
}
```

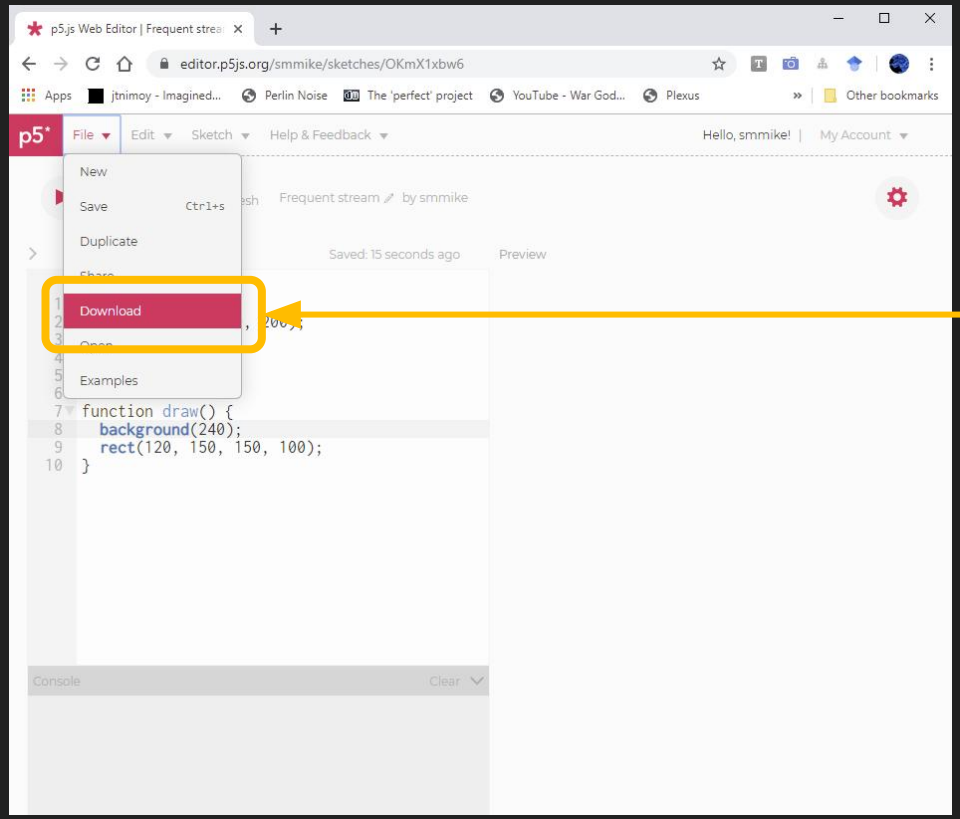
p5.js online editor



Please follow the URL below and create an account so you can save your sketches for later use.

https://editor.p5js.org/

p5.js online editor



Once you have saved your sketch, you may download the sketch as a zipped archive for later user or submission.

A basic p5.js

```
function setup() {
  createCanvas(200, 200);
}

function draw() {
  background(220);
}
```

setup()

runs ONCE ONLY

A basic p5.js

```
function setup() {
  createCanvas(200, 200);
}
```

setup()

runs ONCE ONLY

```
function draw() {
  background(220);
}
```

draw()

LOOPS FOREVER

Our FIRST p5.js Sketch

p5*

p5* File ▾ Edit ▾ Sketch ▾ Help ▾ Log in or Sign up

Auto-refresh Nice rifle

sketch.js

```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(220);
  rect(120, 150, 150, 100);
}
```

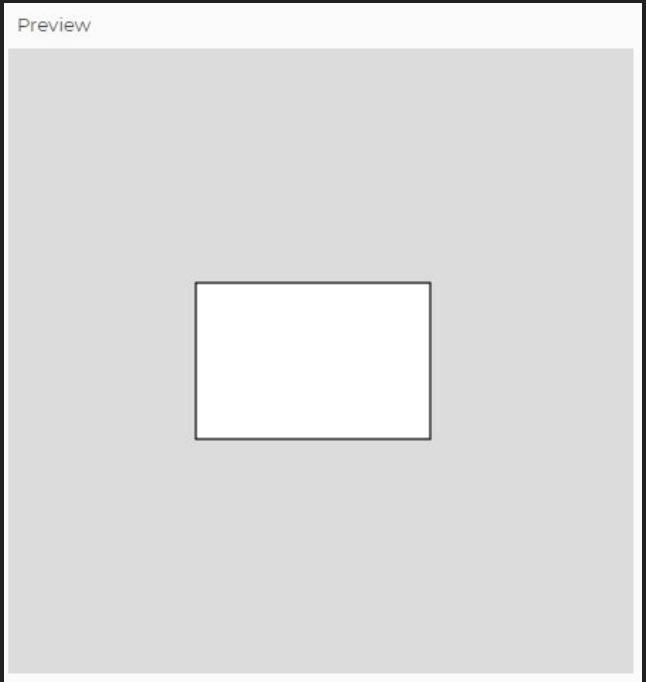
Console Clear

Preview

Our FIRST p5.js Sketch

```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(220);
  rect(120, 150, 150, 100);
}
```

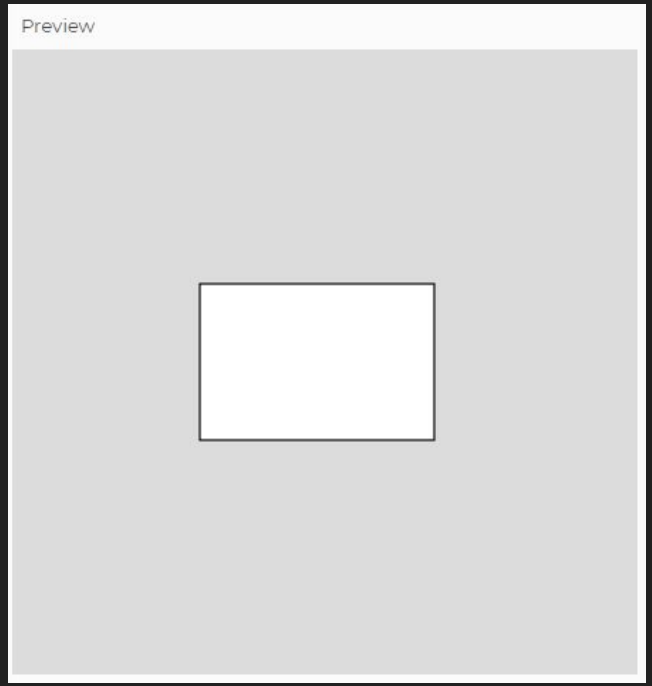


Our FIRST p5.js Sketch

```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(220);
  rect(120, 150, 150, 100);
}
```

To 'create' a drawing area ('canvas') of 400 by 400 pixels.



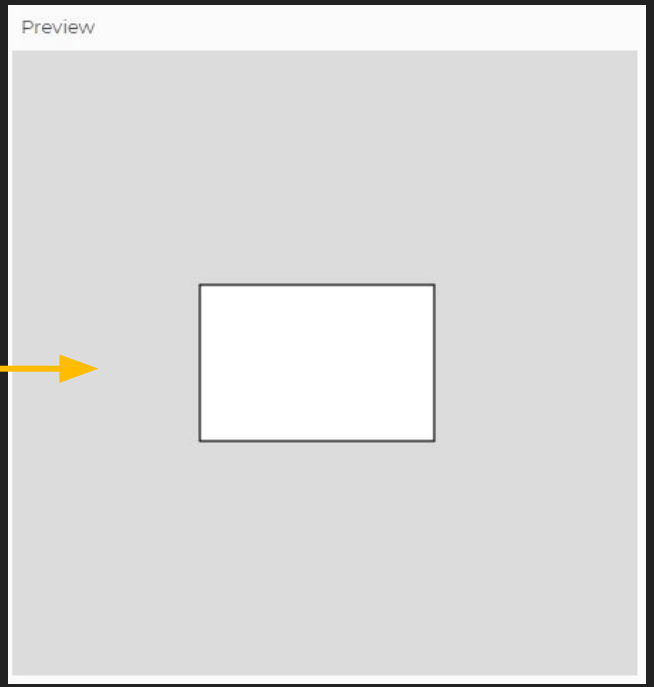
Our FIRST p5.js Sketch

```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(220);
  rect(120, 150, 150, 100);
}
```

To 'create' a drawing area ('canvas') of 400 by 400 pixels.

To 'clear' the background area with a grayscale shade of 220.



Our FIRST p5.js Sketch

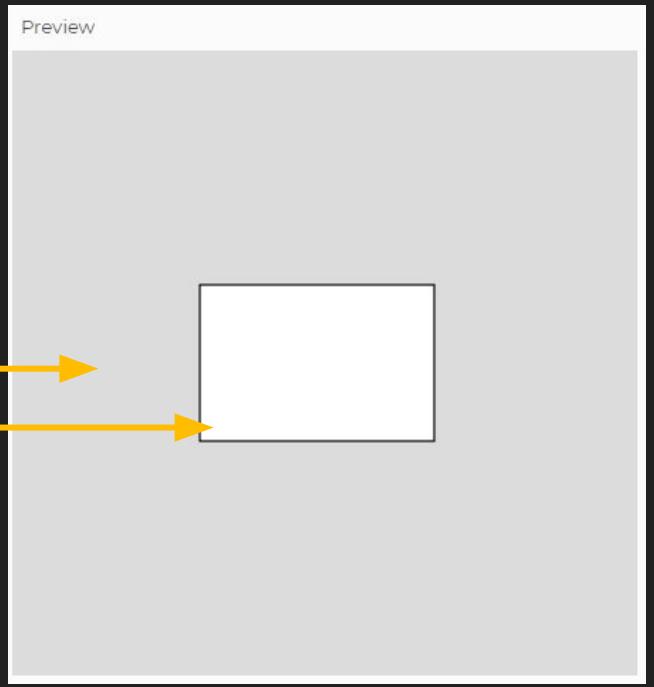
```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(220);
  rect(120, 150, 150, 100);
}
```

To 'create' a drawing area ('canvas') of 400 by 400 pixels.

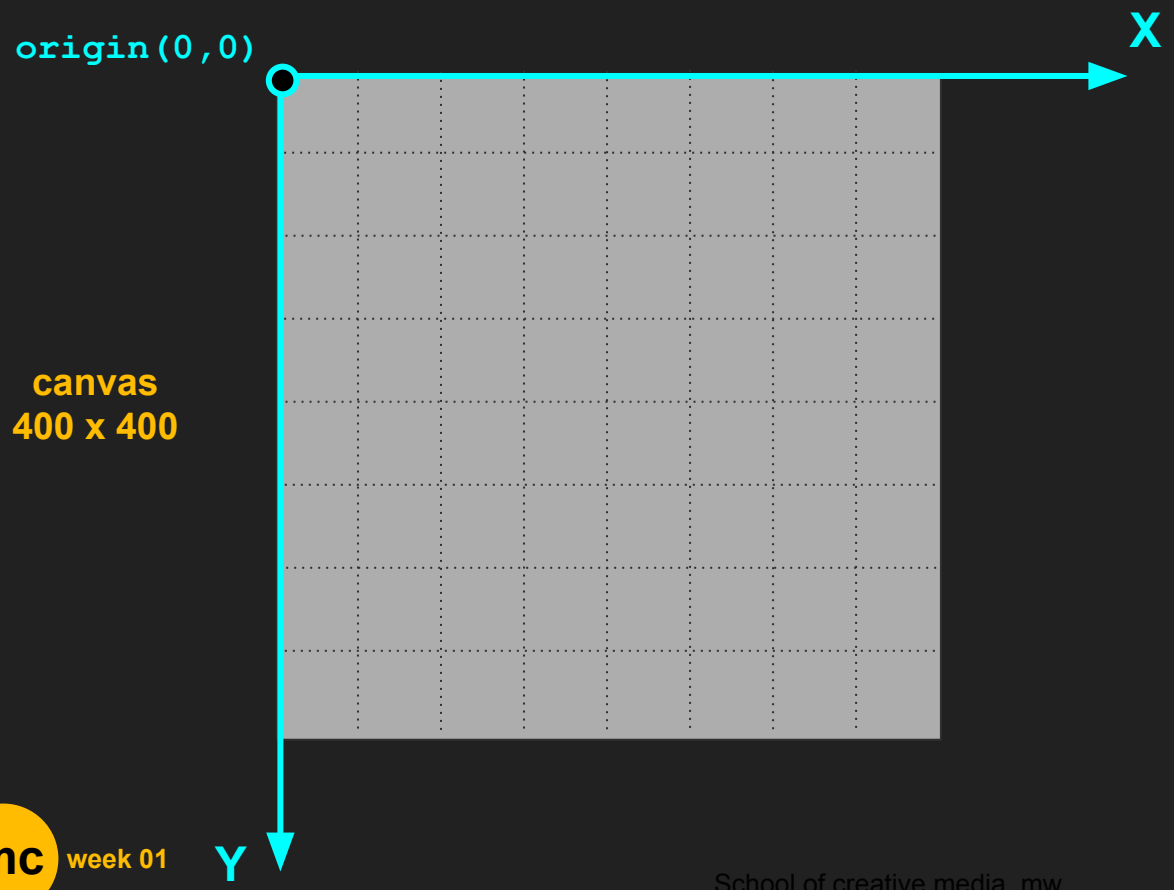
To 'clear' the background area with a grayscale shade of 220.

To 'draw' a rectangle With its upper corner at `(120,150)`; its width and height equal to `(150,100)`.



Drawing Coordinate System in p5.js

p5.js coordinate system



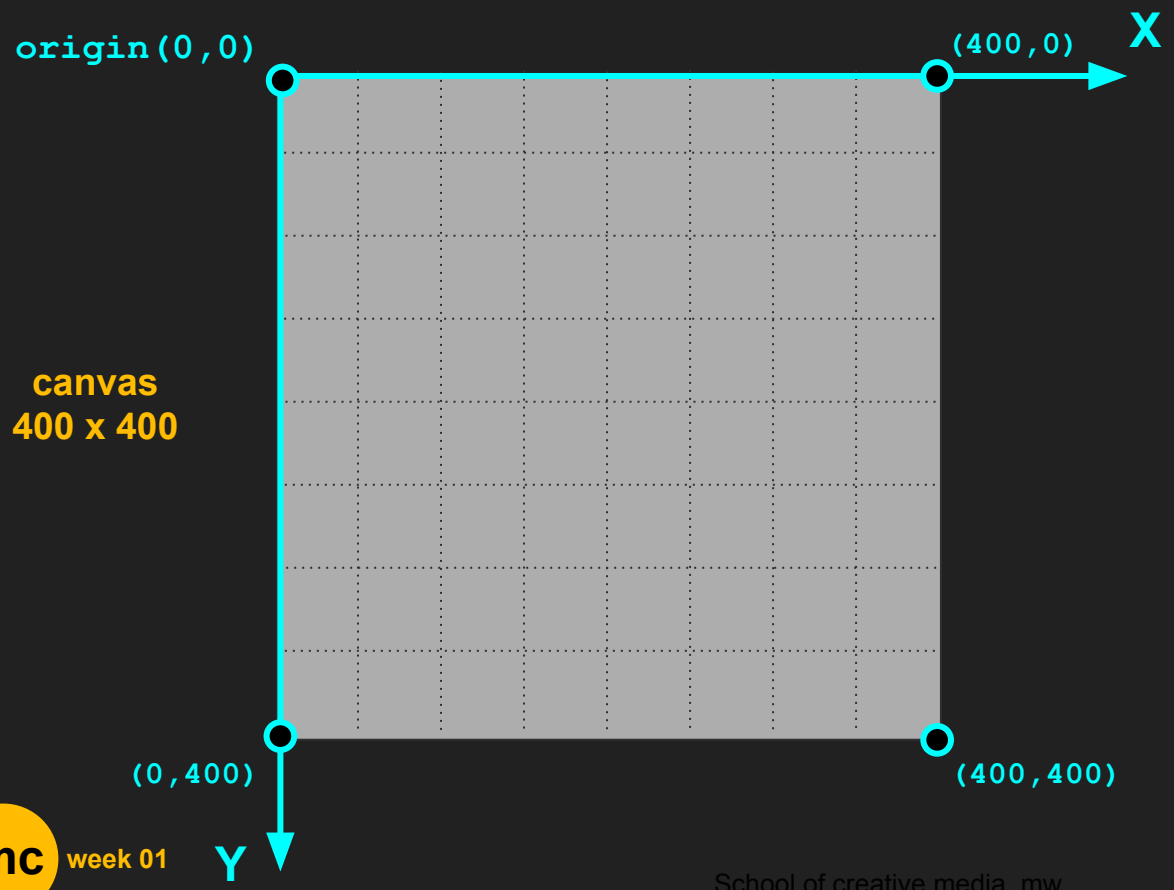
Position of a POINT on a p5.js canvas is defined by a pair of numbers which we call a `coordinate(x,y)`

`x` = horizontal distance from the origin

`y` = vertical distance from the origin

The `origin (0,0)` of a p5.js canvas is at the UPPER LEFT CORNER.

p5.js coordinate system



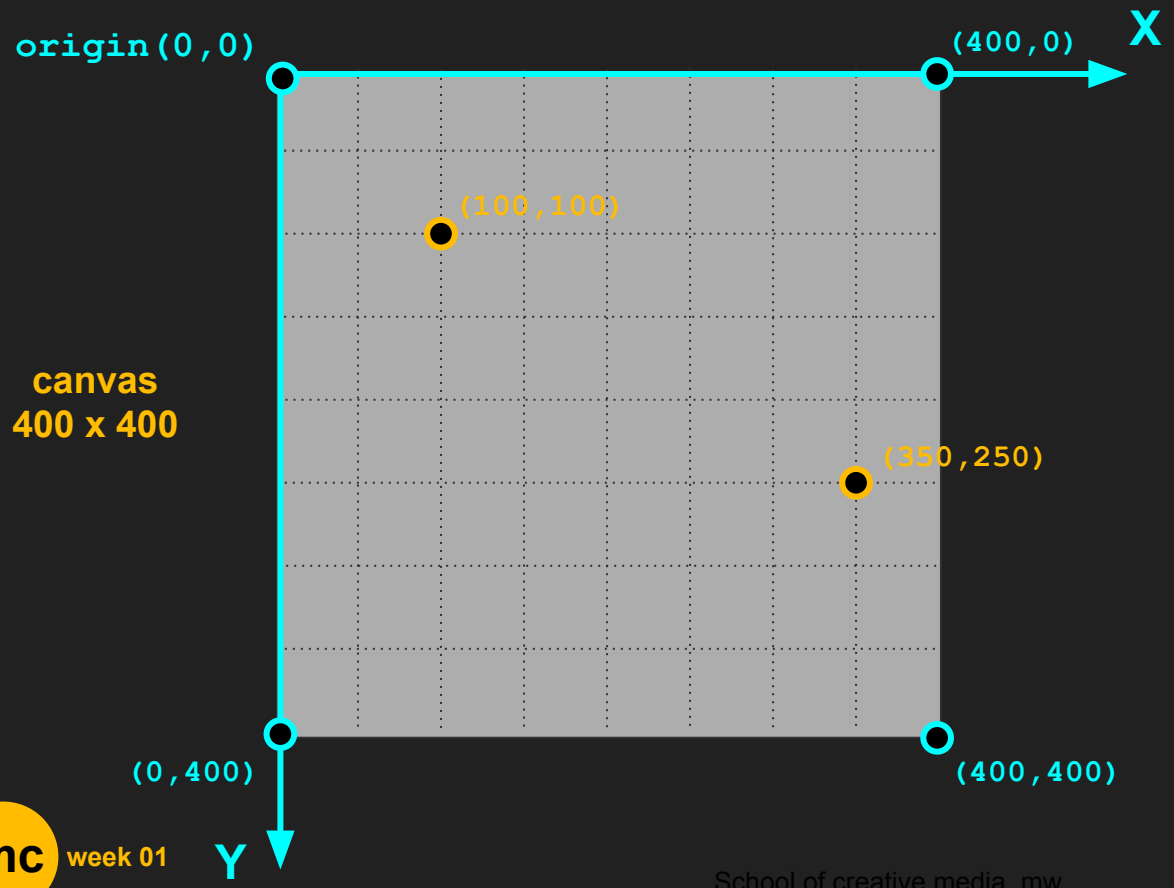
Position of a POINT on a p5.js canvas is defined by a pair of numbers which we call a `coordinate(x,y)`

`x` = horizontal distance from the origin

`y` = vertical distance from the origin

The `origin (0,0)` of a p5.js canvas is at the UPPER LEFT CORNER.

p5.js coordinate system



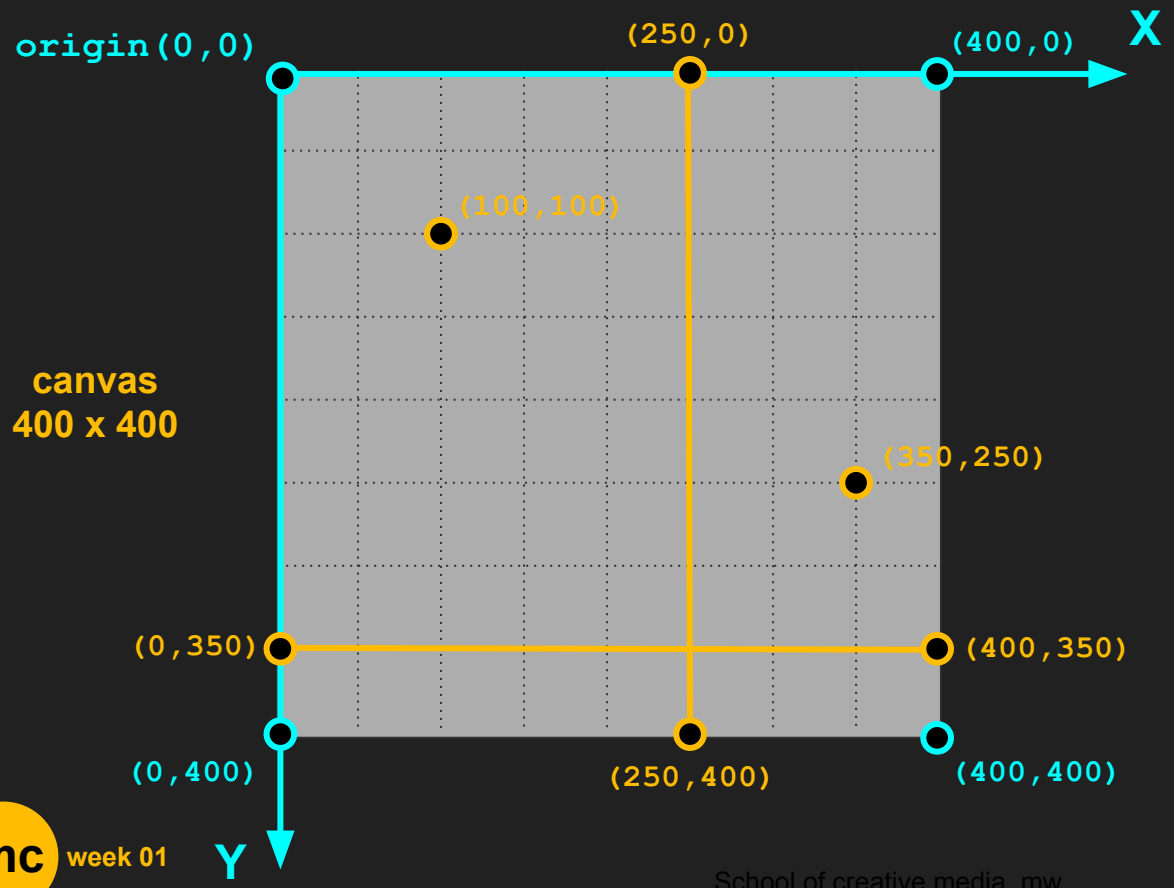
Position of a POINT on a p5.js canvas is defined by a pair of numbers which we call a `coordinate(x,y)`

`x` = horizontal distance from the origin

`y` = vertical distance from the origin

The `origin (0,0)` of a p5.js canvas is at the UPPER LEFT CORNER.

p5.js coordinate system



Position of a POINT on a p5.js canvas is defined by a pair of numbers which we call a `coordinate(x,y)`

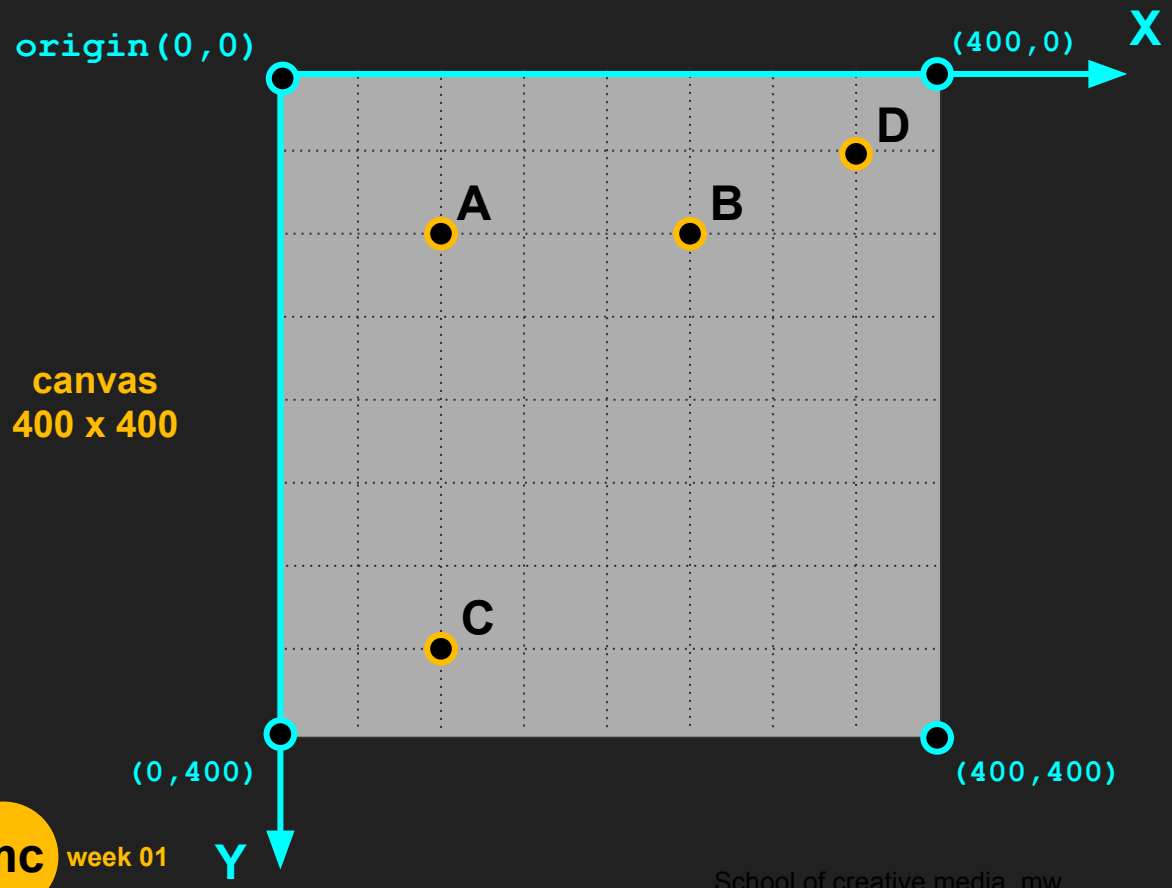
`x` = horizontal distance from the origin

`y` = vertical distance from the origin

The `origin (0,0)` of a p5.js canvas is at the UPPER LEFT CORNER.

p5.js Basic 2D drawing functions

point(x,y);



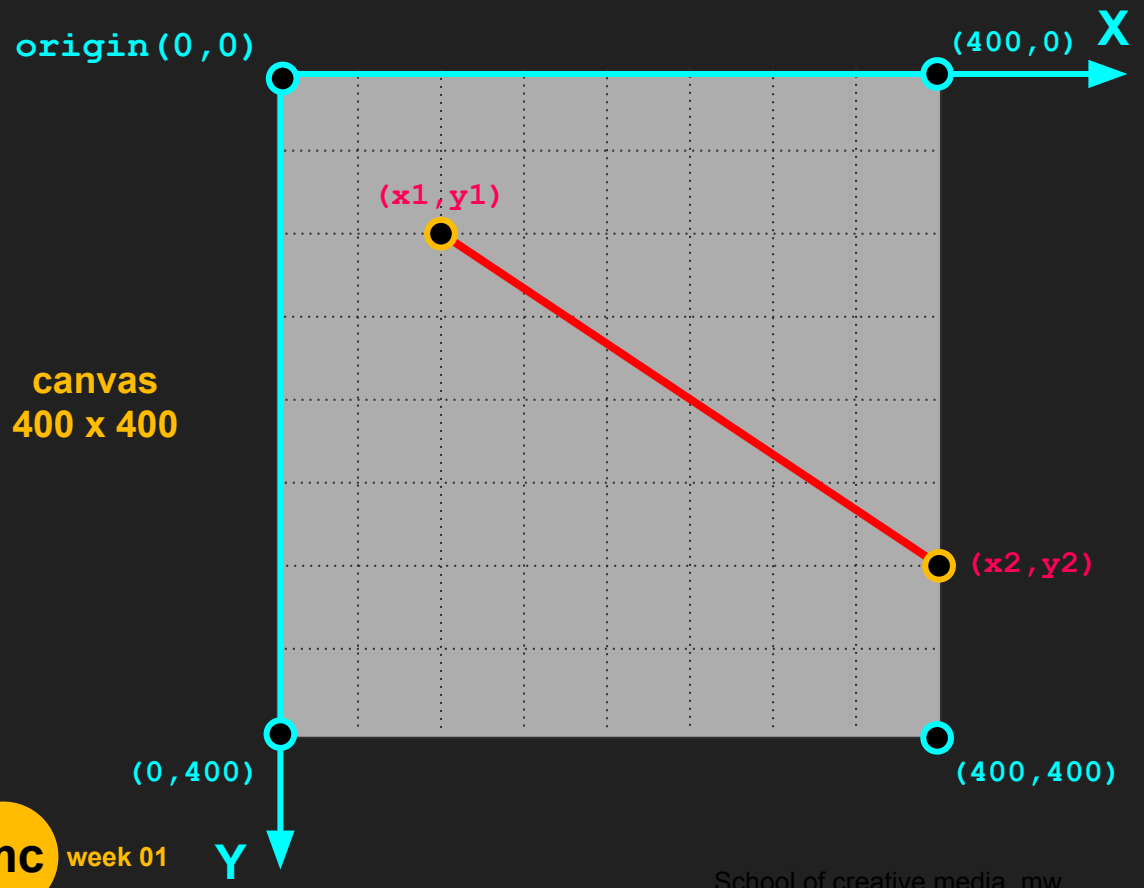
point(x,y)

Draws a one-pixel point at the given position (x,y).

Example:

```
point(100, 100);  // point A
point(250, 100);  // point B
point(100, 350);  // point C
point(350, 50);   // point D
```

line(x1,y1,x2,y2);



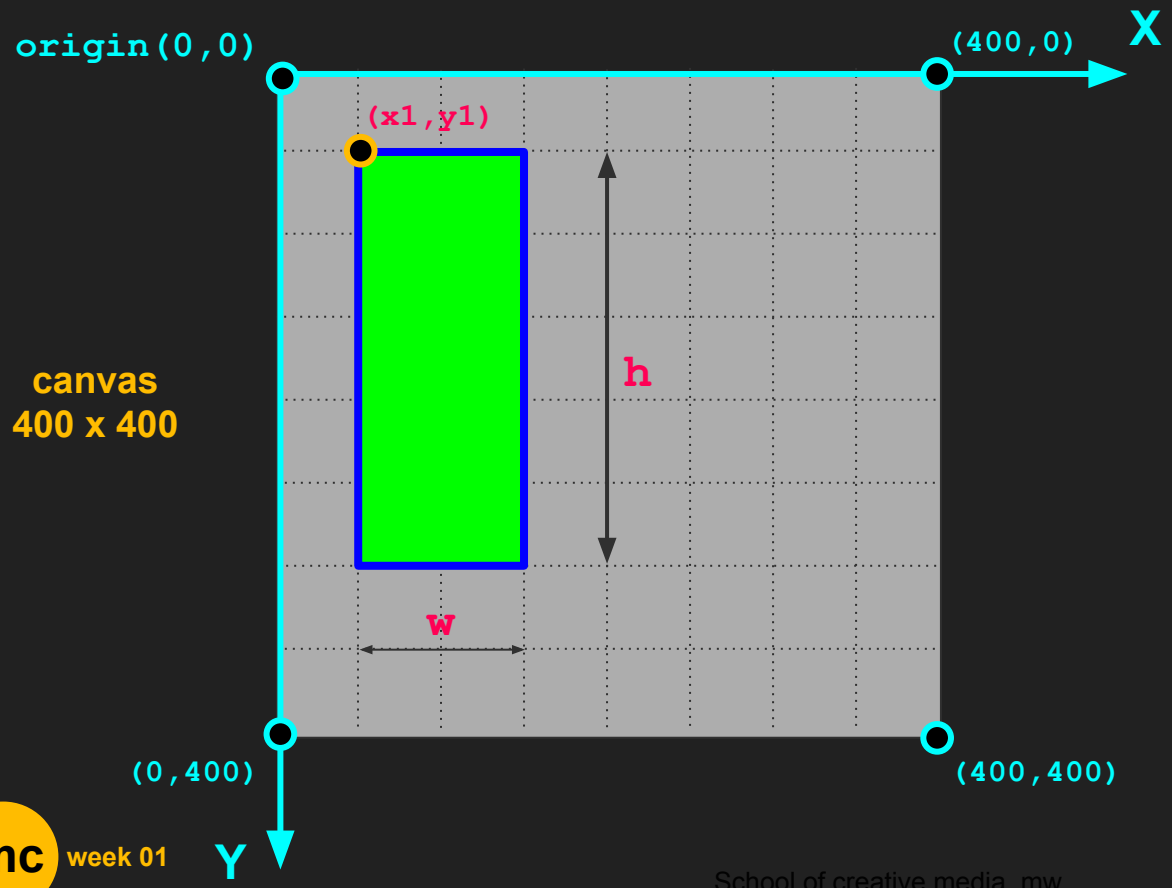
line(x1,y1,x2,y2)

Draws a line defined by two points, (x1,y1) and (x2,y2)

Example:

```
line(100, 100, 400, 300);
```

rect(x1,y1,w,h,[r]);



rect(x1,y1,w,h,[r]);

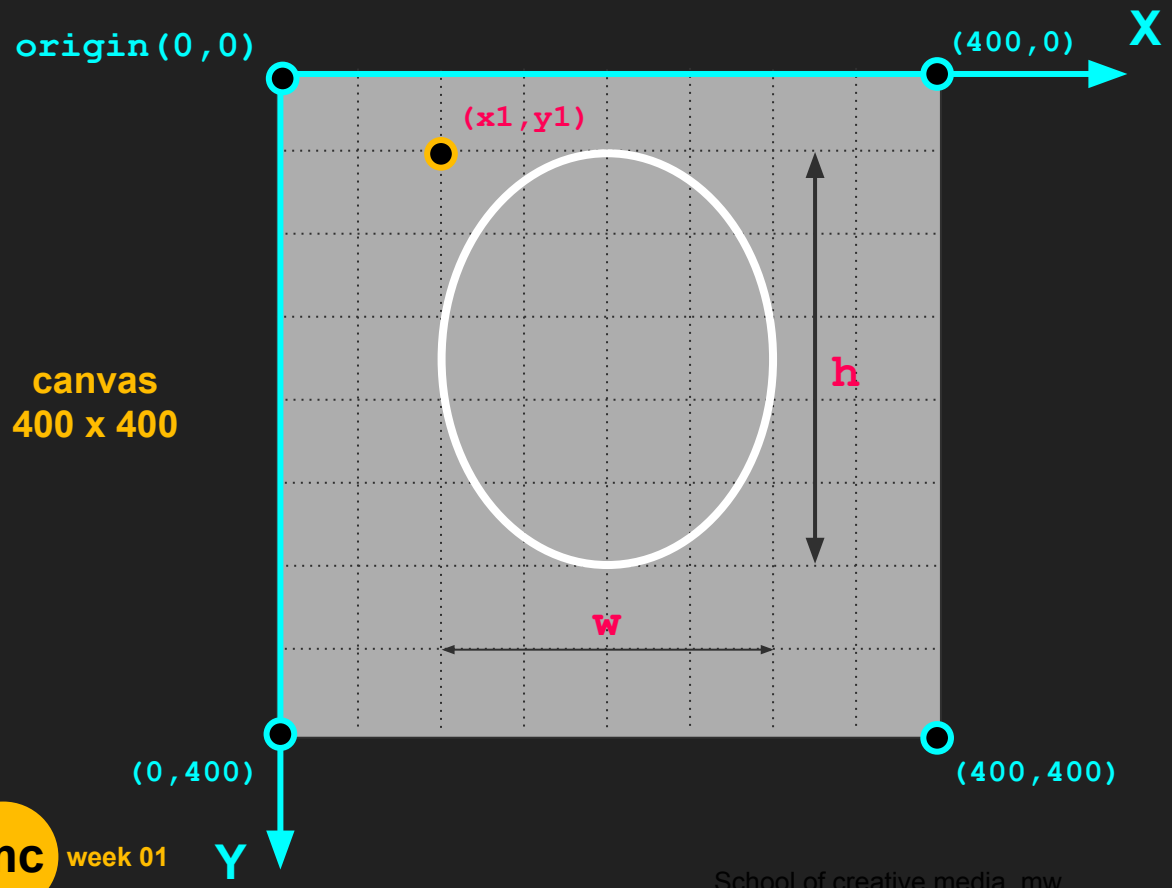
Draws a rectangle defined by a location point* (x1,y1), and the size (w,h) where w = width, and h = height. r is an optional parameter which defines the corner roundness radius in pixel.

location point: **upper left corner**

Example:

```
rect(50, 50, 100, 250);
```

ellipse(x1,y1,w,[h]);



ellipse(x1,y1,w,[h]);

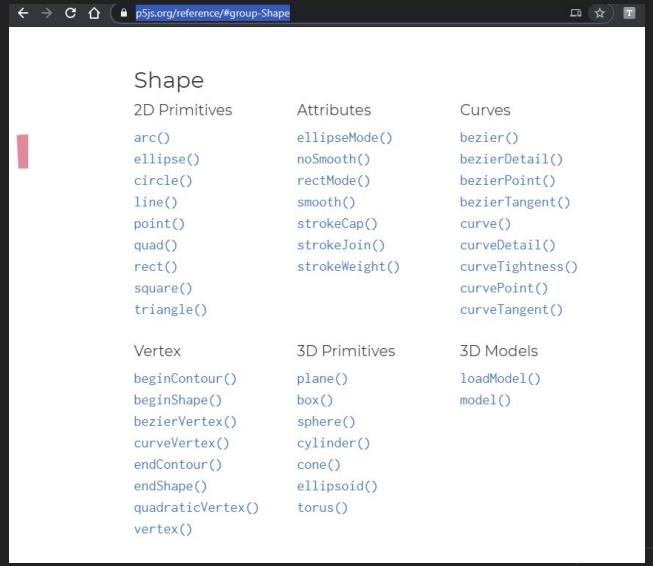
Draws a ellipse defined by a location point* (x1,y1), and the size (w,h) where w = width, and h = height. If h is not given, it draws a circle of width w.

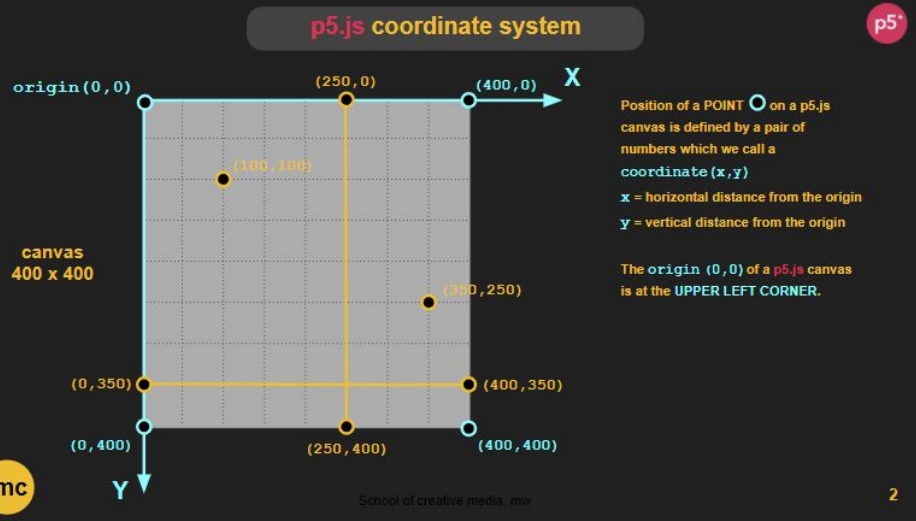
location point: **upper left corner**

Example:

```
ellipse(100, 50, 200, 250);
```

p5.js Shapes References





p5.js program statement

A typical program statement

```
rect(10,10, 200,100);
```

p5.js program statement

a p5.js function

A typical program statement

```
rect(10,10, 200,100);
```

p5.js program statement

a p5.js function

A typical program statement

```
rect(10,10, 200,100);
```

function parameters

p5.js program statement

p5*

each program statement must be terminated with a semicolon ' ; '

a p5.js function

A typical program statement

```
rect(10,10, 200,100);
```

function parameters

p5.js Stroke and Fill Colors

p5.js stroke and fill colors

p5*

p5* File ▾ Edit ▾ Sketch ▾ Help ▾ Log in or Sign up

Auto-refresh Nice rifle

sketch.js

```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(220);
  stroke(0);
  fill(255,0,0);
  rect(120, 150, 150, 100);
}
```

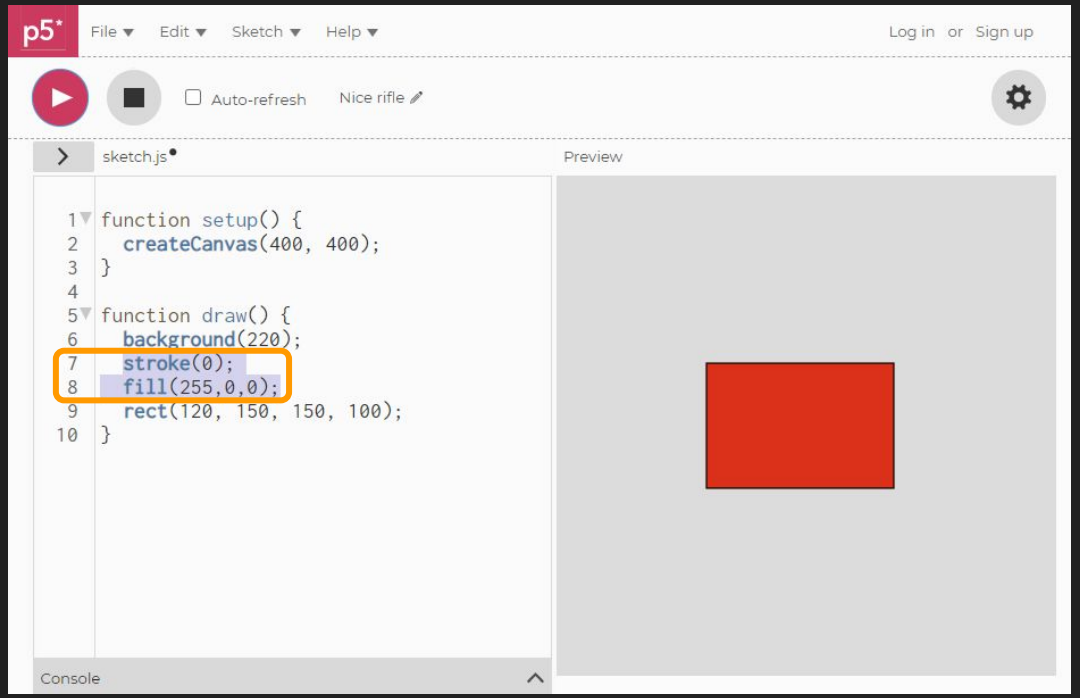
Preview

Console

p5.js stroke and fill colors

```
stroke(0);
```

instructs p5.js to use BLACK (0) as outline color.



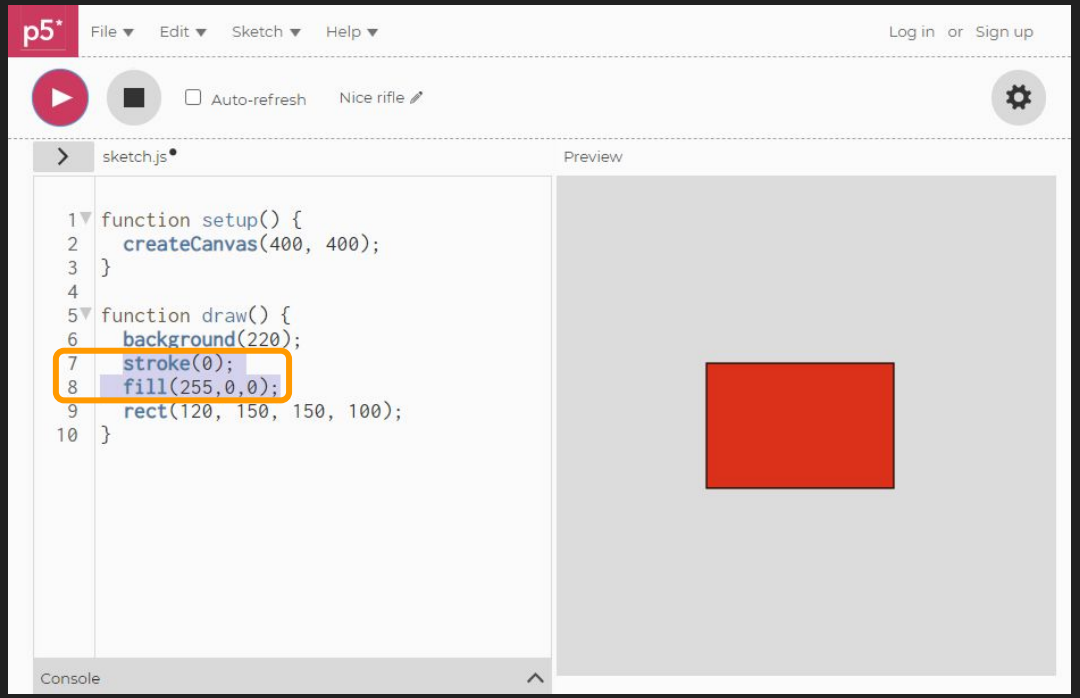
p5.js stroke and fill colors

```
stroke(0);
```

instructs p5.js to use BLACK (0) as outline color.

```
fill(255,0,0);
```

instructs p5.js to use RED (255,0,0) as fill color.



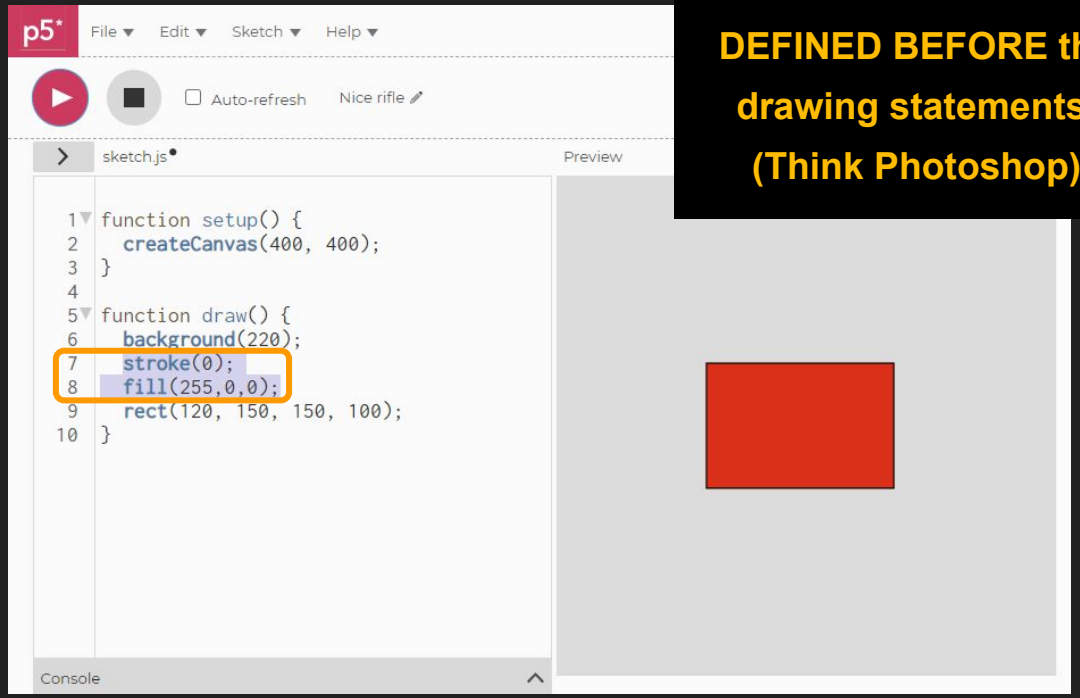
p5.js stroke and fill colors

```
stroke(0);
```

instructs p5.js to use BLACK (0) as outline color.

```
fill(255,0,0);
```

instructs p5.js to use RED (255,0,0) as fill color.



Colors MUST BE DEFINED BEFORE the drawing statements. (Think Photoshop)

p5.js stroke and fill colors

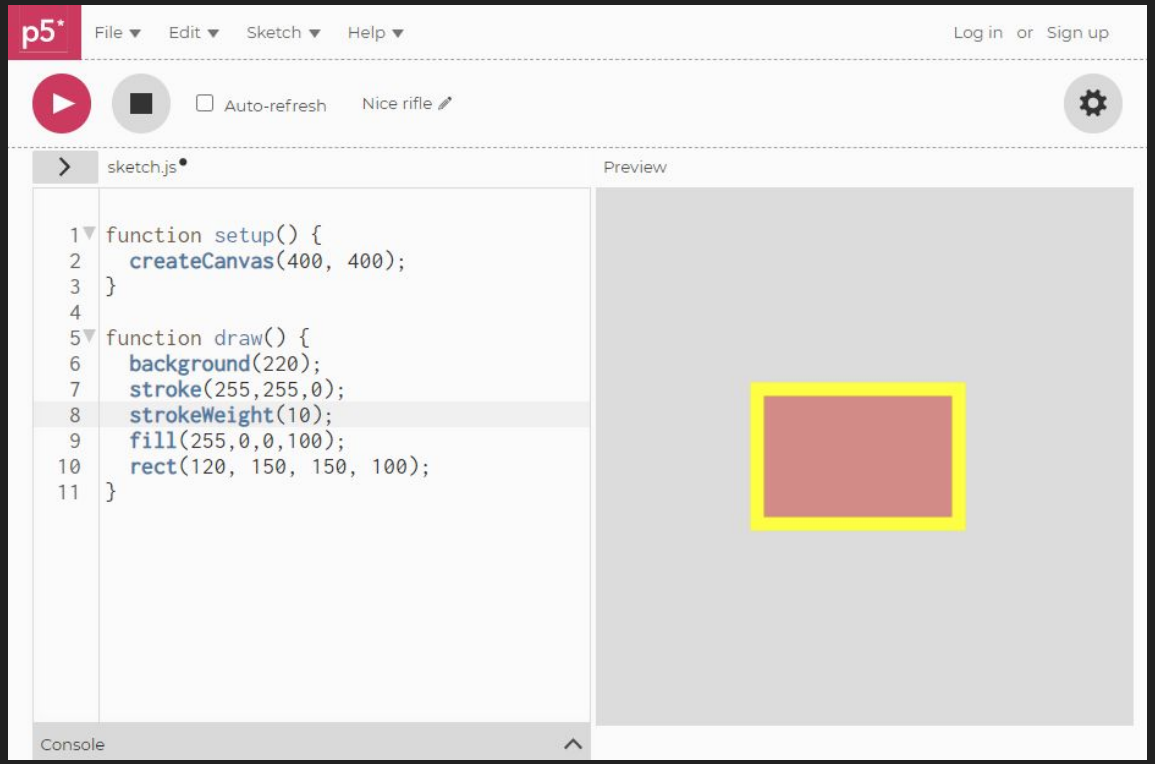
```
stroke(0);              // Grayscale 0-255
stroke(0,255,0);        // R,G,B (0-255)
stroke(0,255,0,100);    // R,G,B,A (0-255)
```

```
fill(255);              // Grayscale 0-255
fill(255,255,0);        // R,G,B (0-255)
fill(0,255,0,100);      // R,G,B,A (0-255)
```

p5.js strokeWeight()

```
strokeWeight(10);
```

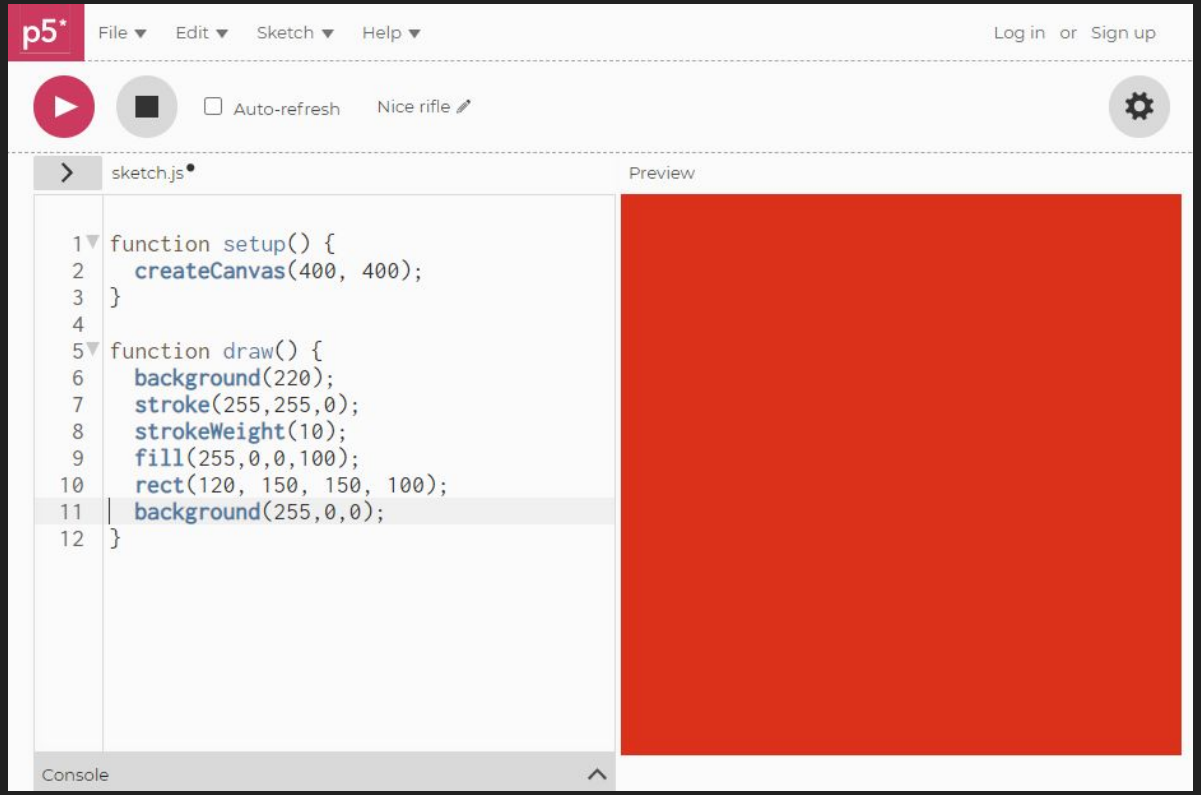
instructs p5.js to set the outline thickness to 10 pixels.



p5.js background()

```
background(255,0,0);
```

instructs p5.js to clear everything on the canvas and fill it with the given color



Built-in function random()

- random() serves as a convenient means to make up some numbers in an arbitrary manner.
- random() takes 0, 1 or 2 parameters.

```
random(); // a random number from 0 to 0.999

random(10); // a random # from 0 to 9.999

random(-10, 10);  // a random # from -10 to 9.999
```

A basic p5.js

```
function setup() {
  createCanvas(200, 200);
}
```

setup()

runs ONCE ONLY

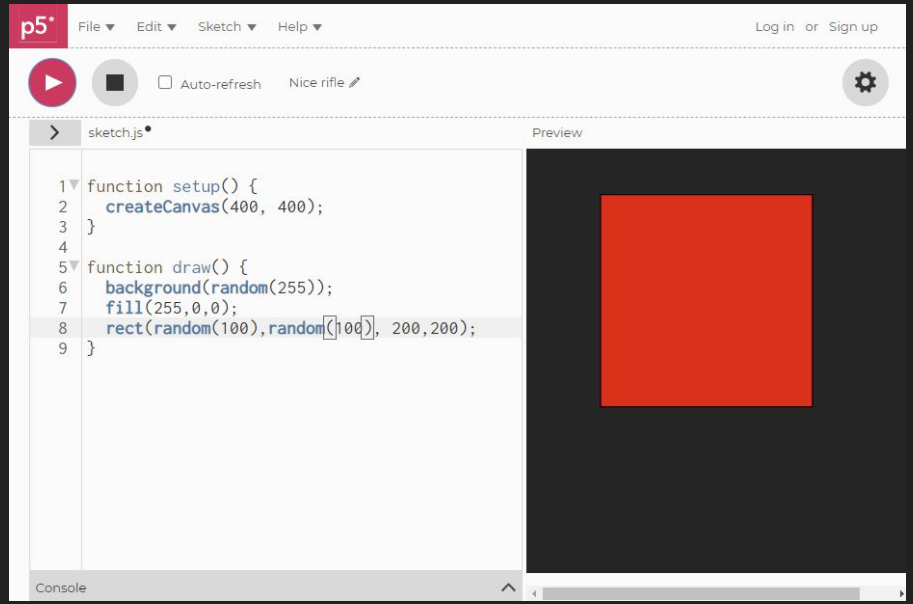
```
function draw() {
  background(220);
}
```

draw()

LOOPS FOREVER

exercise

In-class Exercise



1. Use the `random()` function to generate colors for `stroke()`, `background()` and `fill()` to get changing colors.

2. Use the `random()` function to generate coordinates or size for various 2D shapes.

https://p5js.org/reference/

p5.js Reference

p5*

p5.org/reference/

Processing | p5.js | Processing.py | Processing for Android | Processing for Pi

p5*js

Home
Editor
Download
Donate
Get Started
Reference
Libraries
Learn
Examples
Books

Community
Showcase
Forum
GitHub

Reference

Search reference

Can't find what you're looking for? You may want to check out p5.sound.
You can also download an offline version of the reference.

Color	Environment	Image	Shape
Constants	Events	Lights, Camera	Structure
DOM	Foundation	Math	Transform
Data	IO	Rendering	Typography

Color

Creating & Reading	Setting
alpha()	background()
blue()	clear()
brightness()	colorMode()
color()	fill()
green()	noFill()
hue()	noStroke()
lerpColor()	stroke()