



Let's make HTML5 & JavaScript Games!



credit: w3.org

Before we begin making the game, we're going to go over what each building block will contribute to your game.

HTML is a markup language

- Markup languages are code-based annotation systems
- HTML is the backbone of every website

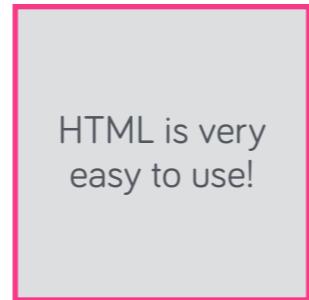
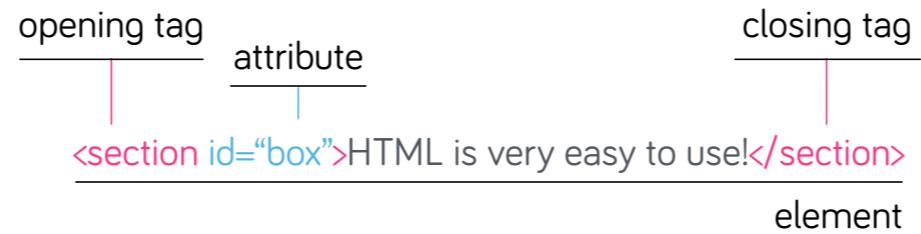
idiosyncratic power. Brown and Duguid's contribution has given studies, ~~still relatively little developed,~~ that seek to understand situated activity.

The assumptions on which innovation may be considered as a consequence of situated in work practices are the following:

- Knowledge is produced through participation in a set of practices
- Participation in work practices leads to the development of a collective memory
- Participation in a practice ~~entails legitimate participation in the new meanings of those practices~~ and the ethical and aesthetic criteria

You mark-up real documents with a pen to separate and highlight content.. In markup languages, you use code!

HTML is made of elements



HTML defines the structure and layout of a website by using different elements.

Elements are composed of tags and their descriptors, also known as attributes.

Every element has a different purpose, similar to the way that each button in Microsoft Word does something different.

Tags are like sides of a box that help to organize content by type. They tell the browser how the content will be used.

HTML5 is new & dynamic

- HTML5 was designed for all of today's internet-capable devices
- Includes new tags for more types of content
- Check out diveintohtml5.info



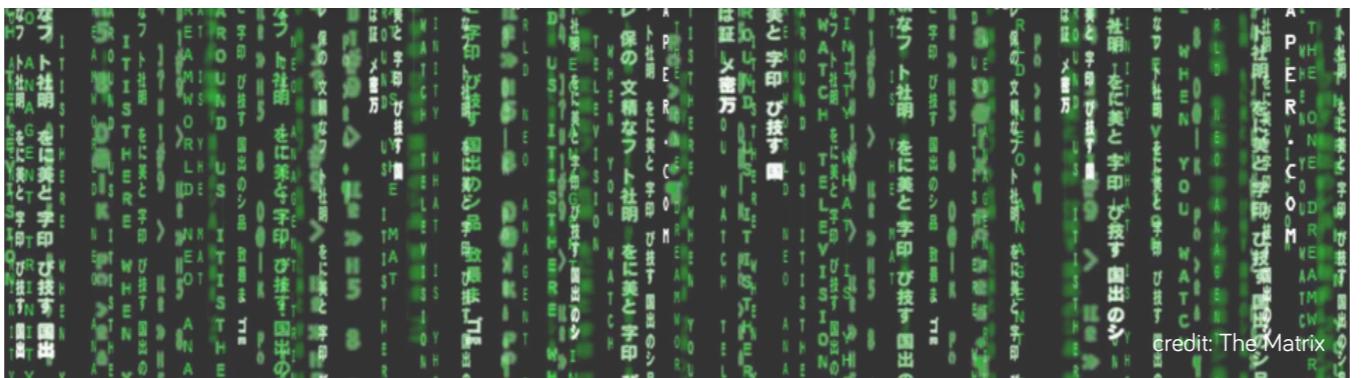
The previous revision of HTML, HTML4, was released in 1997, when almost no one used a cell phone, dial-up was rampant, and no one thought video game consoles or TVs would be able to connect to the internet.

HTML5 is the 5th revision of HTML -- it has been created to keep up with the demand for more dynamic content on the web. Because it's new, it isn't supported by older versions of browsers (like IE8)

It includes new elements, including the one we're going to use to make games today!



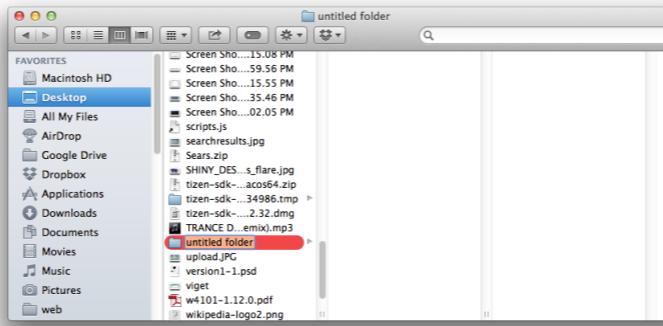
Let's make a simple HTML5 page



credit: The Matrix

We're going to actually start typing. Prepare yourselves.

Create a folder to work in

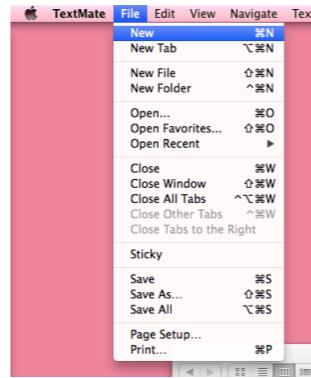


Name your folder **CLF-html5-game**

You can create a new folder on your desktop (or anywhere on your computer that you'd like to store this file)

Name it **CLF-html5-game**.

Create a new file in your text editor



Save it as [index.html](#) in your folder

Go into your text editor and create a new file. Save it in your folder as index.html.

Index.html is a naming convention — on the internet it's the first file the browser looks for when loading a website

Save as often as possible; occasionally computers are dumb. I'm scarred from Photoshop, so I save my files very often!

Components of an HTML5 page

- `<!DOCTYPE html>`: tells the browser it is looking at an HTML5 page
- `<html>`: begins the HTML code
 - `<head>`: the area where meta information will be located
 - `<meta charset="utf-8">`: the character encoding you want to use
 - `<title>`: the website title
 - `<body>`: the part of the page where we will be working!

To begin, you have to tell the browser what kind of page it is looking at. There are several doctypes; some are for older versions of HTML and some are for XML or SVG.

Next, we tell the browser it's time to read HTML! I may use the word "parse" sometimes when referring to the browser reading code.

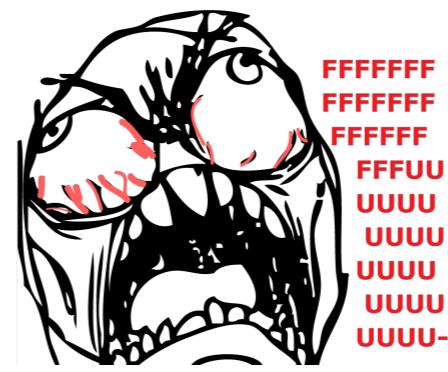
The HTML tag must be closed. The head is where we put all the information(data) that will be seen by search engines.

Comment your code

5 minutes after you write code
without comments



When you come back to it
in 3 weeks



You should always take notes in your code so you remember what it does.

Commenting code is easy

- Preface your comment with <, a bang and two dashes (`<!--`)
- End it with two dashes and > (`-->`)
- Most text editors have shortcuts (like  + /)



```
index.html
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <title>My cool HTML5 game</title>
6  </head>
7  <body>
8      <!-- This is a comment -->
9      This isn't a comment!
10 </body>
11 </html>
```

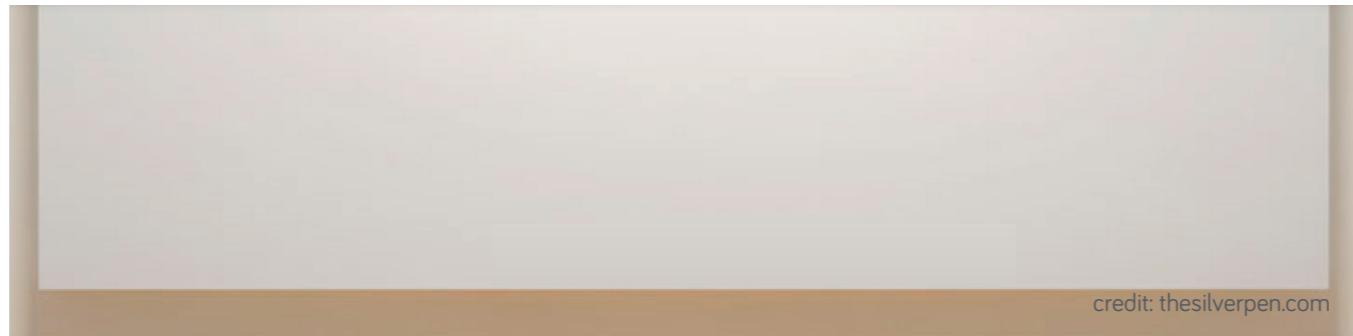
If you open index.html in your browser, you should see “This isn’t a comment!”, but not “This is a comment”.

Does anyone have questions so far?

Is everyone set up with a basic web page?



The canvas element



The canvas element is exactly what it sounds like — a dynamic artboard for both interactive animations and games.

What is it?

A canvas is a rectangle in your page where you can use JavaScript to draw anything you want.



credit: Nintendo

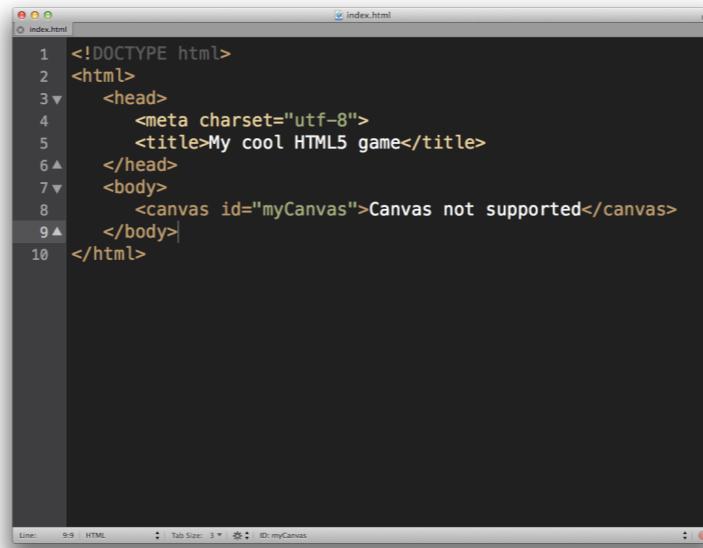
Once iPhones and Android dropped support for Flash, the canvas element began to look more appealing as a way to spread content across web-enabled devices.



credit: Photon Storm

Once iPhones and Android dropped support for Flash, the canvas element began to look more appealing as a way to spread content across web-enabled devices.

Create a <canvas> element



```
index.html
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <title>My cool HTML5 game</title>
6 </head>
7 <body>
8   <canvas id="myCanvas">Canvas not supported</canvas>
9 </body>
10 </html>
```

To begin, open and close your canvas element.

Add an id so you can find it later.



Draw on the canvas with JavaScript



credit: Jandi Small

We're about to go in deep, so please let me know if anything is unclear.

Make a javascript file

Include the javascript file in the HTML below the canvas — must be after the canvas so JS can find it

Is everyone ready?

Variables

- Variables are useful for storing data that **may change** throughout the course of your app (e.g. your player's health, location)
- To create a variable, you have to tell JavaScript:
 - The name you're going to refer to it by
 - The value (information) that the variable contains

By variables, I am referring to numbers that will change, such as the location of the character and enemy on the screen.

Create an FPS variable for our game loop.

Variables

- Variables let you refer to the same information many times
- If you need to change that information, you only have to do it once

For example, best friends may change but the label stays the same:

```
var myBestFriend = "Isaiah";
var myBestFriend = "Rebecca";
var myBestFriend = "Aileen";
```

By variables, I am referring to numbers that will change, such as the location of the character and enemy on the screen.

Create an FPS variable for our game loop.

Functions

- **Function:** a named section of a program that does a specific task
 - Wraps up code in an easy-to-reference way
 - **Parameter:** additional information you can give the function to change the output

JavaScript has functions, much like a body does. Some functions are included in JavaScript, like `math()` and `random()`, while others must be defined before being used.

- Really all a function is is a named section of a program that does a specific task
- Makes it easier to execute the code you want to execute
- Functions can also take parameters, which is additional info you can give the function to change the output

Function structure

```
var fetch = function (dog) {  
    run to the ball;  
    pick up the ball;  
    bring the ball back;  
};
```

- Name of the function
- Parentheses: Hold any modifiers (also known as arguments)
- Brackets: What to do in the function
- Semicolon: end of line, move onto the next thing

- We will use the first syntax
- Name of function
- Parentheses
- Semicolon: end of line, p. much

Setting up the canvas

- `document`: refers to the HTML document the JavaScript is linked in
- `getElementById`: a native JavaScript function that looks for an ID attribute on the HTML document
- `getContext`: tells JavaScript whether we will make a 2d or 3d drawing
 - The d must be lowercase
- `myCanvas.width`: the width of our canvas
- `myCanvas.height`: the height of our canvas

Calculations

$+$ (add)

$-$ (subtract)

$*$ (multiply)

$/$ (divide)

```
var addition = 13 + 22;  
var division = 100 / 15;
```



Like a flipbook, the canvas animates with frames

With an FPS of 60, the canvas will be redrawn 60 times in one second. It works similarly to a flipbook!

Frames



Frame 1

Frame 6

Framerate and intervals

```
setInterval(function() {  
    do stuff;  
}, 1000);
```

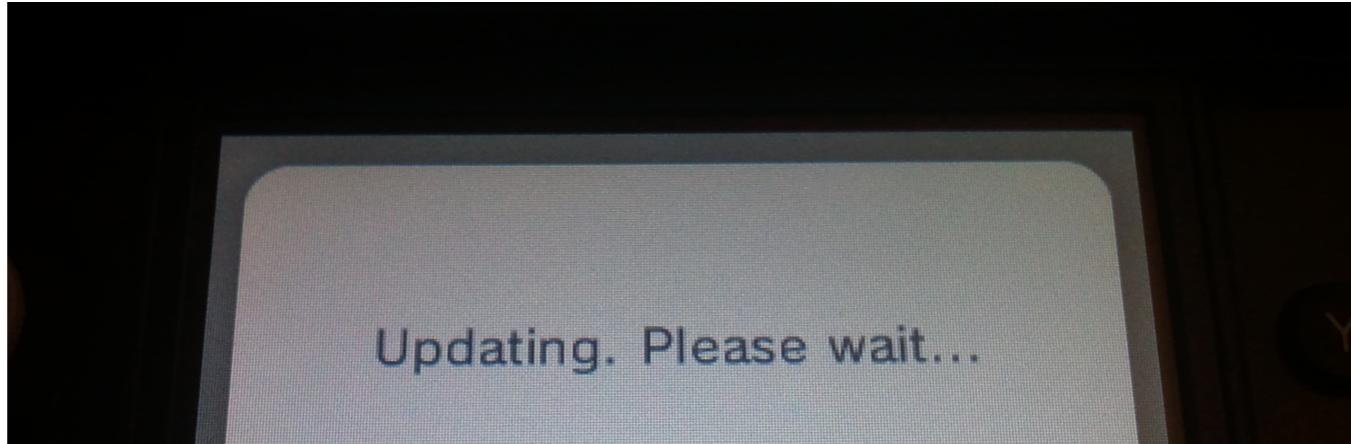
- **setInterval()**: a native JavaScript function that runs a set of code once per set amount of time
 - JavaScript thinks of time in milliseconds
 - 1 second = 1000 milliseconds
- How do we get our canvas to redraw at 60 times in one second?

Framerate and intervals

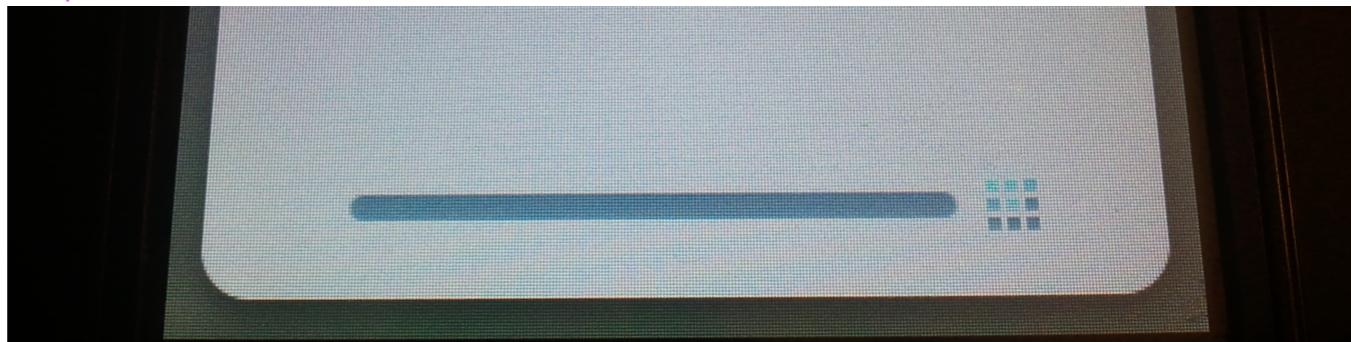
- How do we get our canvas to redraw at 60 times in one second?

1000/60

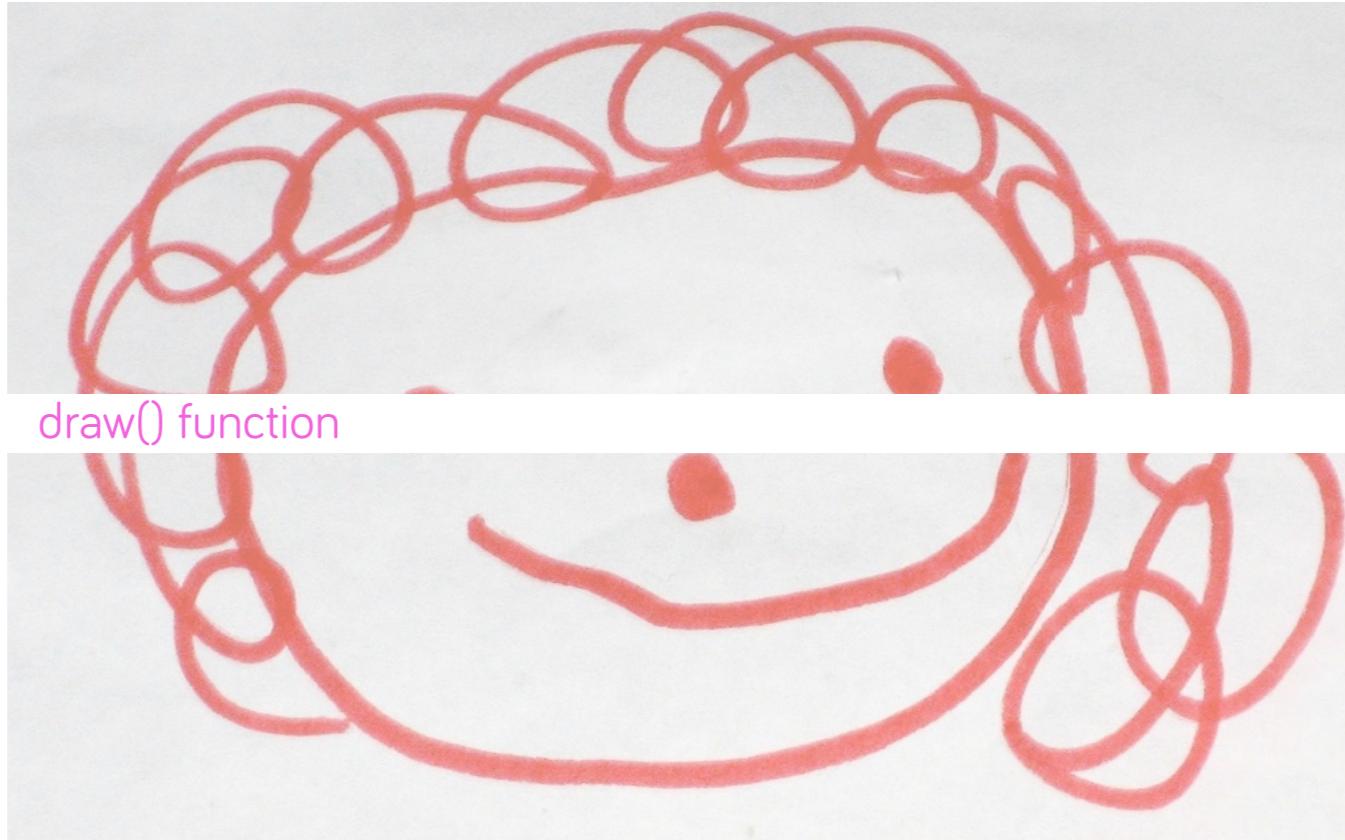
```
setInterval(function() {  
    do stuff;  
}, 1000/60);
```



update() function

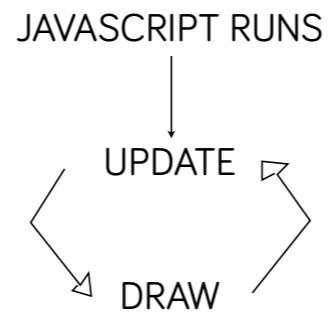


Variables we create will be changed in the update function.



draw() function

- draw()
- Once the update function is finished, the canvas will be redrawn.
- Will draw later shapes over earlier shapes, and use earlier colors for all subsequent shapes (unless you change the color/turn it off)



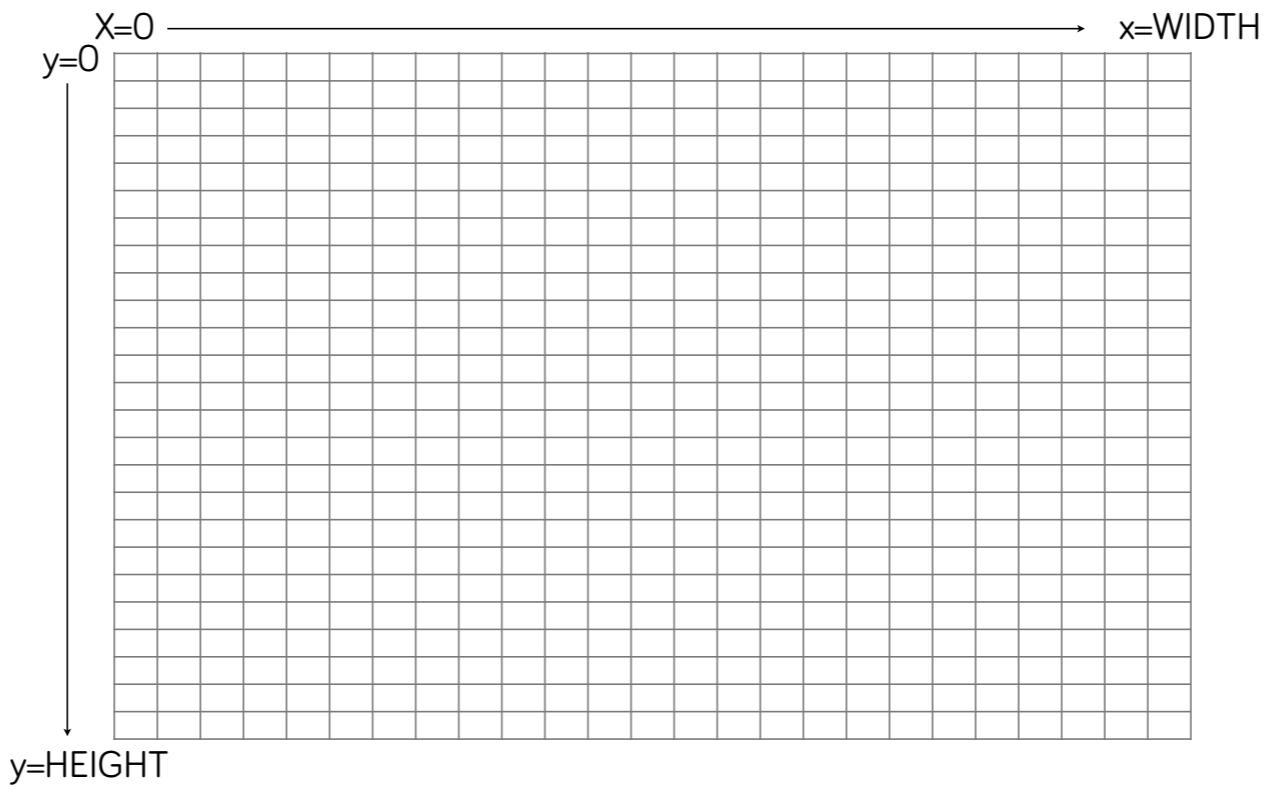
60 times per second

FLOW:

- Javascript begins
- initial variable values are defined
- update executes
- draw executes
- update and draw alternate until the game is finished

How does positioning work?

In order to draw something on screen, you have to tell the computer exactly where to put it.



Positioning

- X: gets bigger as it goes to the right
- Y: gets bigger as it goes down



Let's draw our player at 3,2.

How can I incorporate interactivity?

- Keyboard inputs to move your character

Create a variable object for keys that are currently being pressed. This will allow us to move at diagonals because you can press more than one key.

Event listeners

```
addEventListener("britishAreComing", function (numberOfSoldiers) {  
    paulRevere.ride(horse);  
    town.alert(numberOfSoldiers);  
});
```

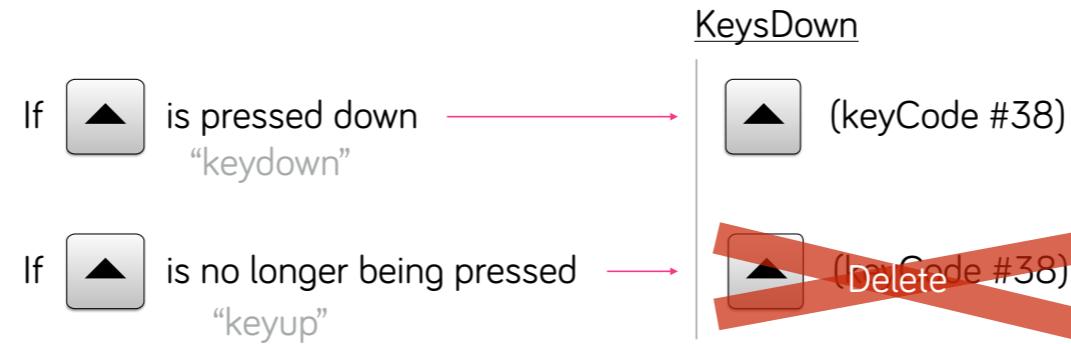
- **addEventListener:** JavaScript waits for something to happen
 - When the event happens, the contained function will run

- Keyboard inputs to move your character

Create a variable object for keys that are currently being pressed. This will allow us to move at diagonals because you can press more than one key.

Next, add an event listener, which will check for

Keyboard interactivity

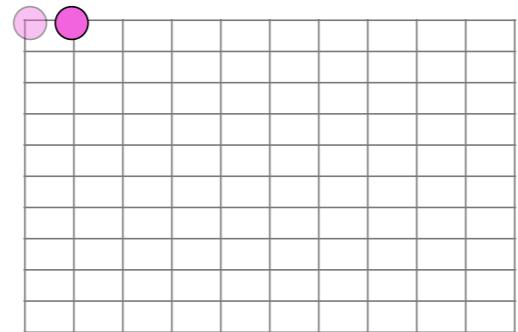


- Keyboard inputs to move your character

Create a variable object for keys that are currently being pressed. This will allow us to move at diagonals because you can press more than one key.

Next, add an event listener, which will check for

Movement

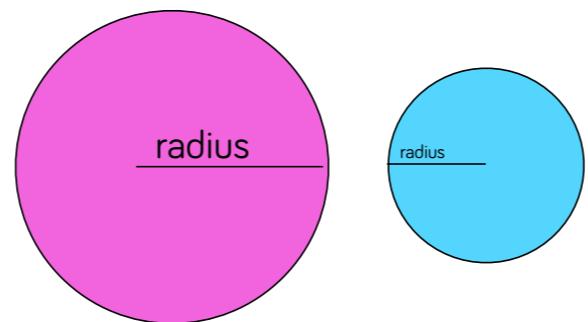


New position = old position + speed

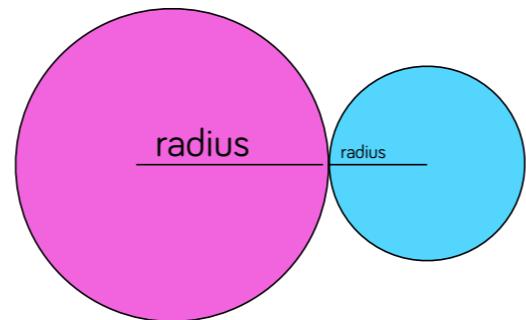
Move our player using the keyboard!

How can I test for collisions?

Have these circles collided yet?



How about now?



If player's X or Y minus the radius equals the location of any edge of the canvas, it must stop moving.

If the player touches the enemy, the enemy must be eaten and the score must increase.

Questions? How is everyone doing

Game: eat the squares!

Randomize the location of all enemies every time one is eaten

Aim to eat all the squares

Keep score

If you eat all of them, you win and the game ends.