NEXGEN CODECAMP

Coding in Javascript

Apr 2018
Instructors:
Jeremy

NEXGEN CODECAMP

Introductions.

What is your name?

What would you like to learn?

NEXGEN CODECAMP

Breaks.
~10:30am

~12:30pm

What we will be doing

NEXGEN CODECAMP

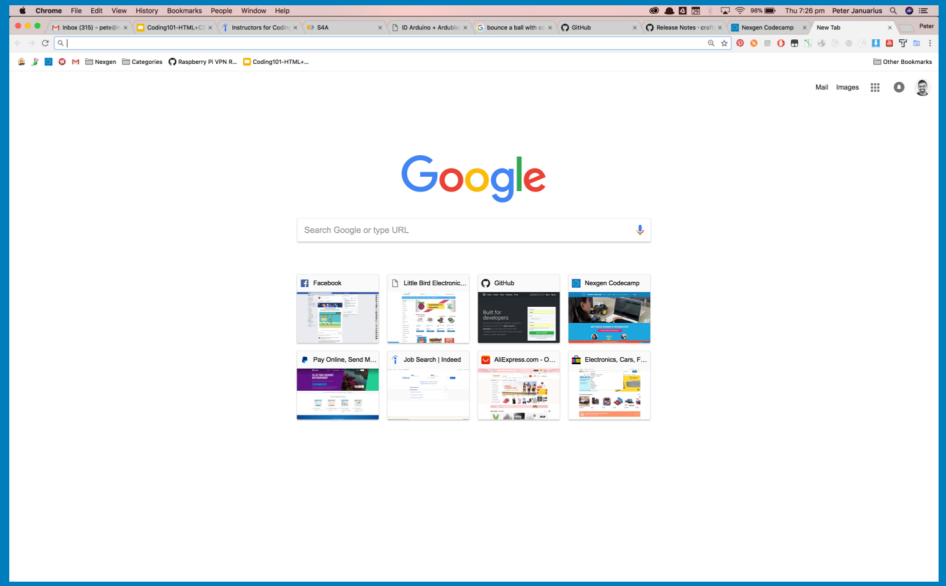
- Learning Javascript
- Basic coding concepts
- Building an endless shooter
- Using our own web servers

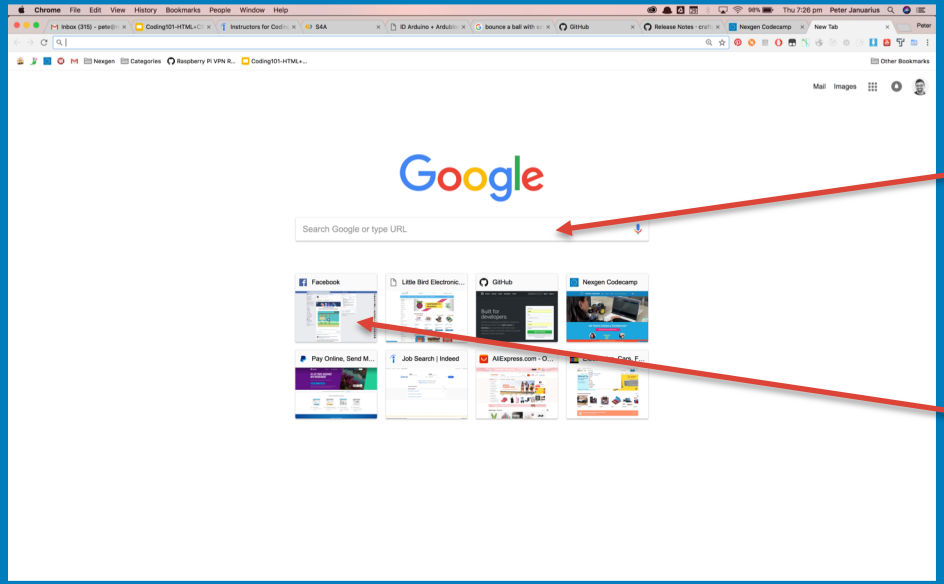
NEXGEN CODECAMP

Lesson 1 - Concepts & Game Intro

NEXGEN CODECAMP

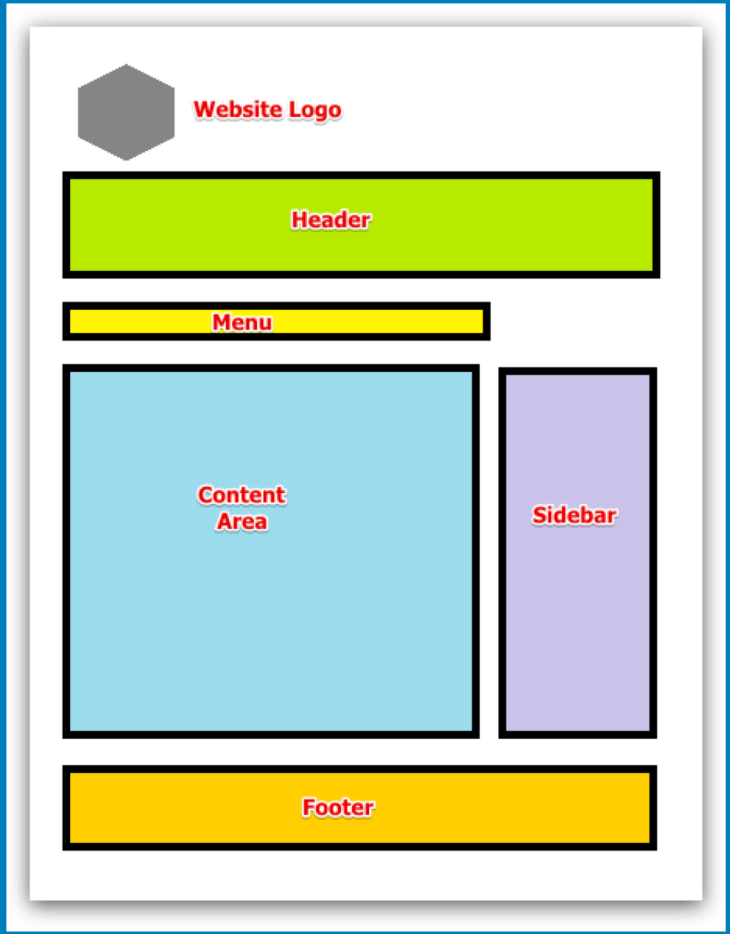
Every webpage is rendered (displayed) by your browser in HTML



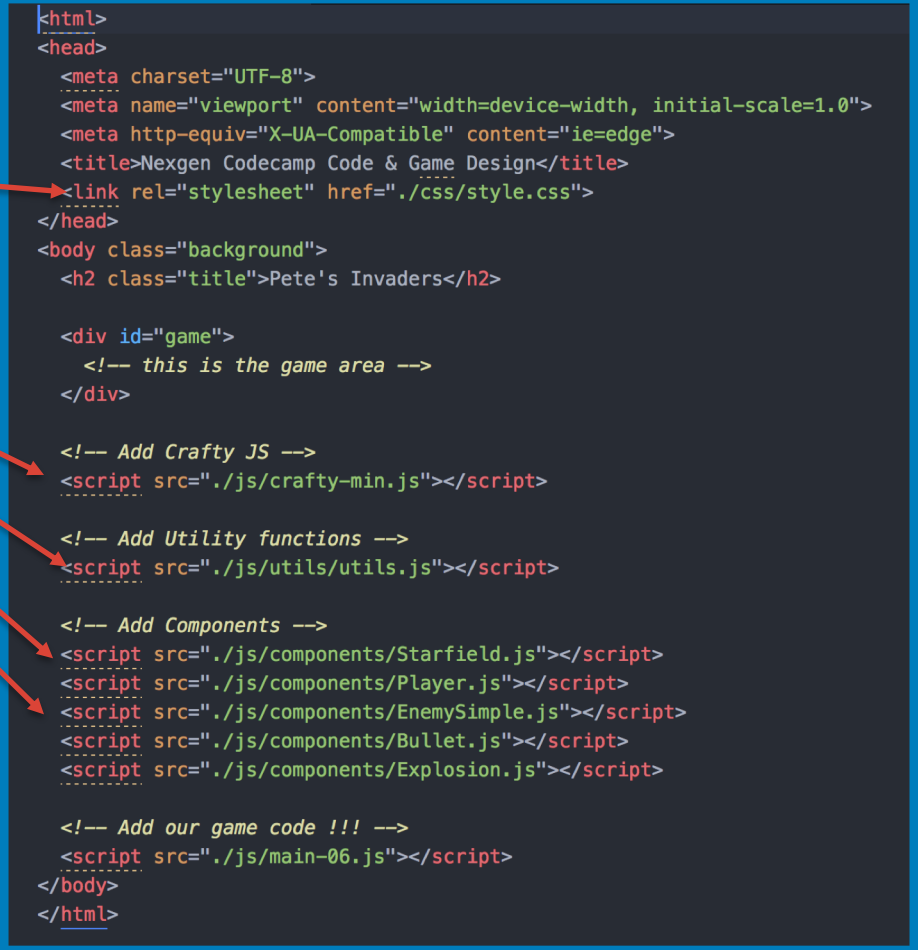



```
<!DOCTYPE html>
<html lang="en-AU">
▶<head>…</head>
▼<body class="default-theme des-mat" style="background: rgb(255, 255, 255);">
 ▶<div class id="mngb">…</div>
 ▼<div id="TZA4S">
  ▶<div class id="lga">…</div>
  ▶<form action="/search" id="f" method="get">…</form>
  ▼<div class="mv-hide" id="most-visited">
   ▼<div id="mv-tiles" style="width: 681px;">
    ▼<iframe id="mv-single" src="chrome-search://most-visited/single.html?removeToo
     ▼#document
       <!DOCTYPE html>
      ▼<html>
         <!-- Copyright 2015 The Chromium Authors. All rights reserved.
              Use of this source code is governed by a BSD-style license that can
              found in the LICENSE file. -->
        ▶<head>…</head>
        ▼<body style="--tile-title-color:#000000;"> == $0
         ▼<div id="most-visited">
          ▼<div id="mv-tiles" style="opacity: 1;">
            ▶<a class="mv-tile" data-tid="1" href="https://www.facebook.com/" ar
            ▶<a class="mv-tile" data-tid="2" href="http://littlebird.com.au/" ar
            Australia Australia - In Stock | Little Bird Electronics" title=
            Australia - In Stock | Little Bird Electronics">…</a>
            ▶<a class="mv-tile" data-tid="3" href="https://github.com/" aria-lab
            ▶<a class="mv-tile" data-tid="4" href="https://nexgencodecamp.com.au
            ▶<a class="mv-tile" data-tid="5" href="http://paypal.com/" aria-labe
            Australia" title="Pay Online, Send Money or Set Up a Merchant Account
            ▶<a class="mv-tile" data-tid="6" href="http://indeed.com/" aria-labe
            ▶<a class="mv-tile" data-tid="7" href="https://www.aliexpress.com/"
            Fashion, Home & Garden, Toys & Sports, Automobiles and More." title="
            Home & Garden, Toys & Sports, Automobiles and More.">…</a>
            ▶<a class="mv-tile" data-tid="8" href="http://ebay.com.au/" aria-lab
            eBay" title="Electronics, Cars, Fashion, Collectibles, Coupons and Mo
            </div>
          </div>
         </body>
        </html>
     </iframe>
    </div>
   </div>
  </div>
 ▶<script nonce="ClHzRDKiOoLDWYPMvtZyyQ==">…</script>
  <script src="/xjs/_/js/k=xjs.ntp.en.bWJr3A6PO-g.O/m=spch/am=gAgSIw/exm=sx,jsa,ntp…c
  async gapi_processed="true"></script>
</body>
```

An HTML page is essentially the structure or the look of the page:



An HTML page also contains links to stylesheets (css) and code (javascript):



```
<html>
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Nexgen Codecamp Code & Game Design</title>
  <link rel="stylesheet" href="./css/style.css">
</head>
<body class="background">
  <h2 class="title">Pete's Invaders</h2>

  <div id="game">
    <!-- this is the game area -->
  </div>

  <!-- Add Crafty JS -->
  <script src="./js/crafty-min.js"></script>

  <!-- Add Utility functions -->
  <script src="./js/utils/utils.js"></script>

  <!-- Add Components -->
  <script src="./js/components/Starfield.js"></script>
  <script src="./js/components/Player.js"></script>
  <script src="./js/components/EnemySimple.js"></script>
  <script src="./js/components/Bullet.js"></script>
  <script src="./js/components/Explosion.js"></script>

  <!-- Add our game code !!! -->
  <script src="./js/main-06.js"></script>
</body>
</html>
```

NEXGEN CODECAMP

JavaScript is a cross-platform, object-oriented scripting language used to make webpages interactive

Mac/Windows/ Linux

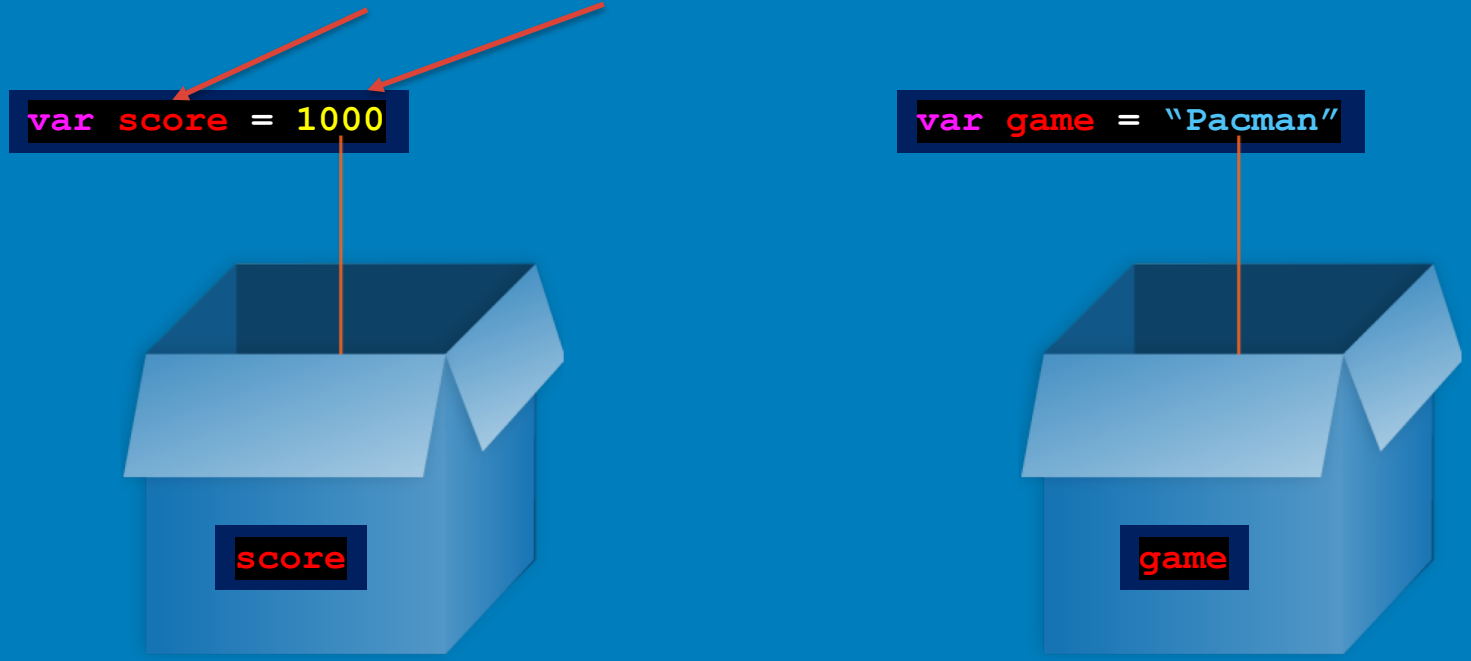
System built with Objects

Mouse over, Click, Keypress

NEXGEN CODECAMP

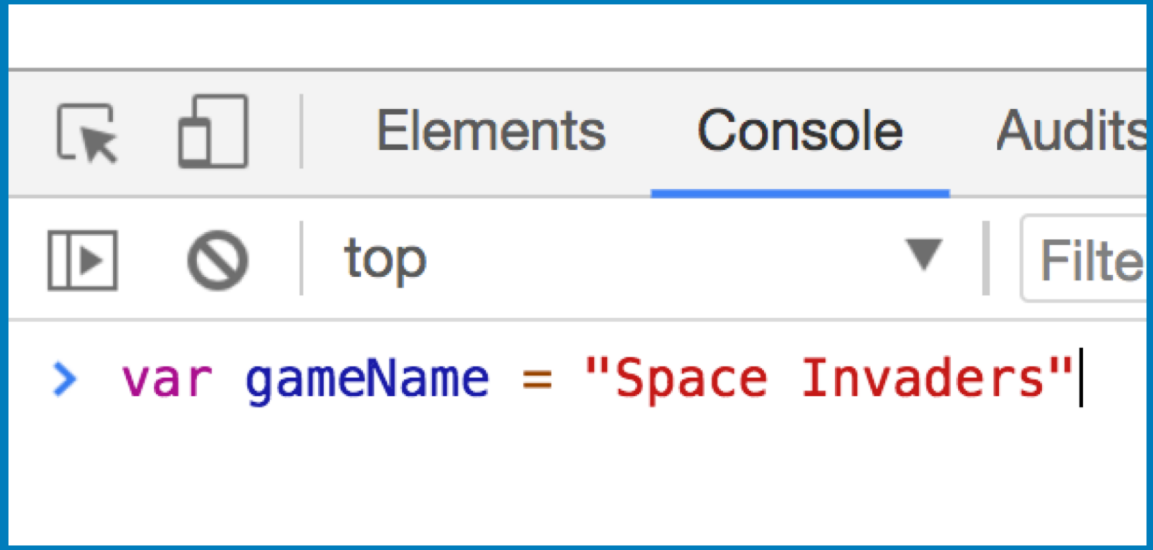
Variables (containers)

Variables are used to store data/information such as **numbers** or **characters**
Variables have names and values



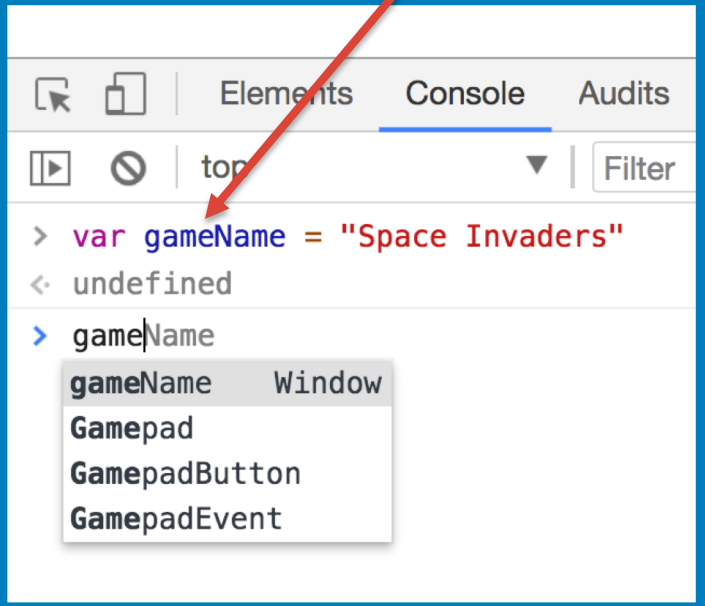
NEXGEN CODECAMP

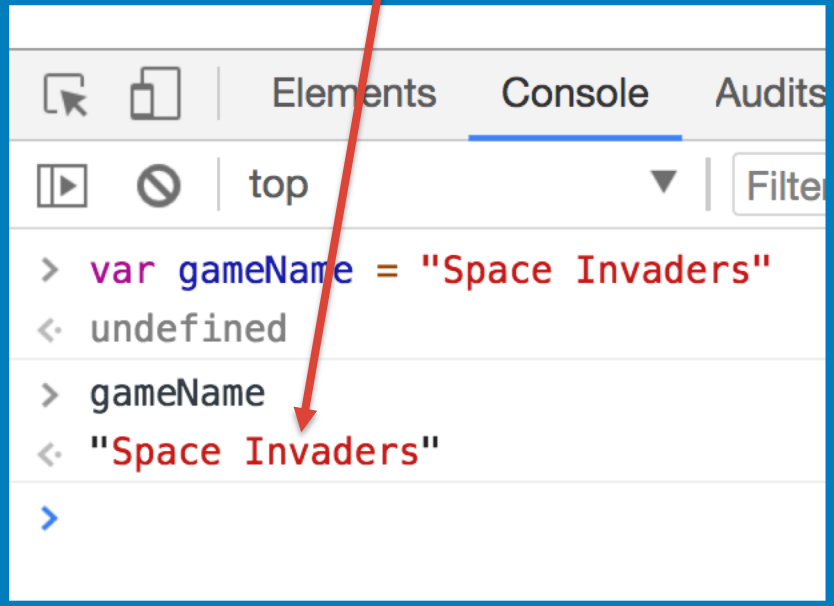
1. Open Google Chrome
2. Use Ctrl+Shift+I (or Cmd+Opt+I on Mac) to open the DevTools.
3. Click on the ‘Console’ tab
4. Create a variable of your choice and hit enter to add it to Chrome



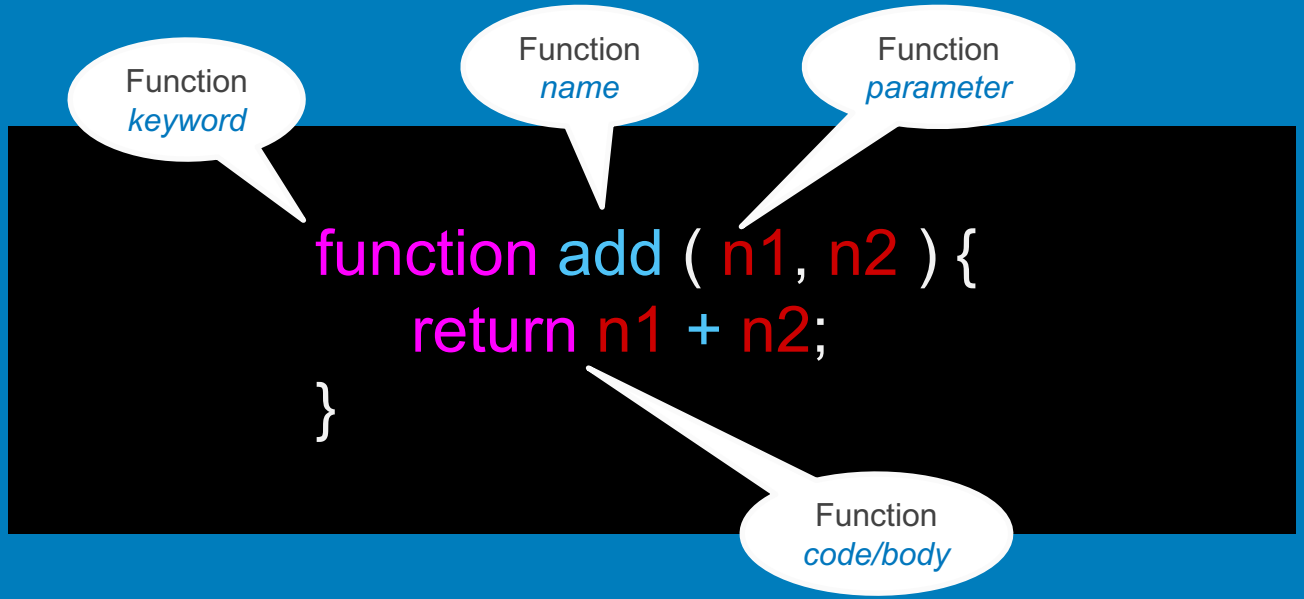
NEXGEN CODECAMP

1. Type in the variable name. You will see that Chrome has it stored
2. Hit the Enter key. Chrome will display the variable value

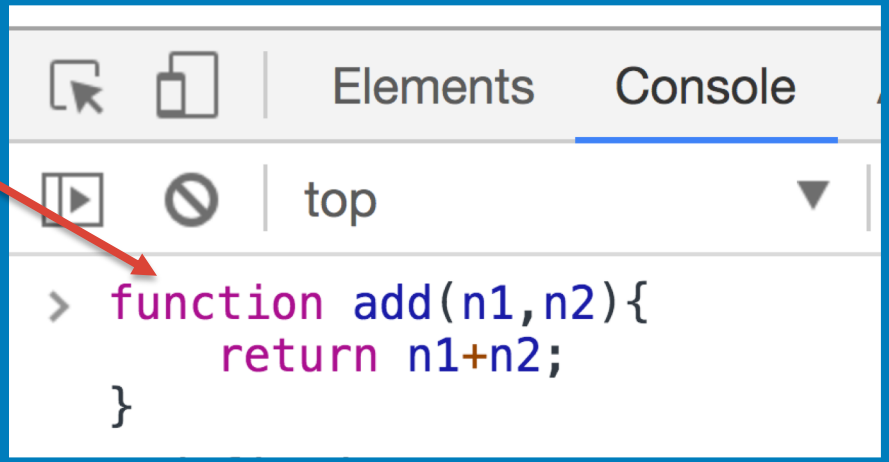




Function (actions)



Create the ‘add’ function in Chrome



Once created, call the function, passing two numbers to add:

```
add(2, 3)
```

NEXGEN CODECAMP

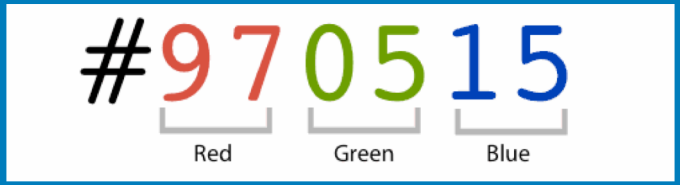
HEX Colors

#BE0712

#1E9C5A

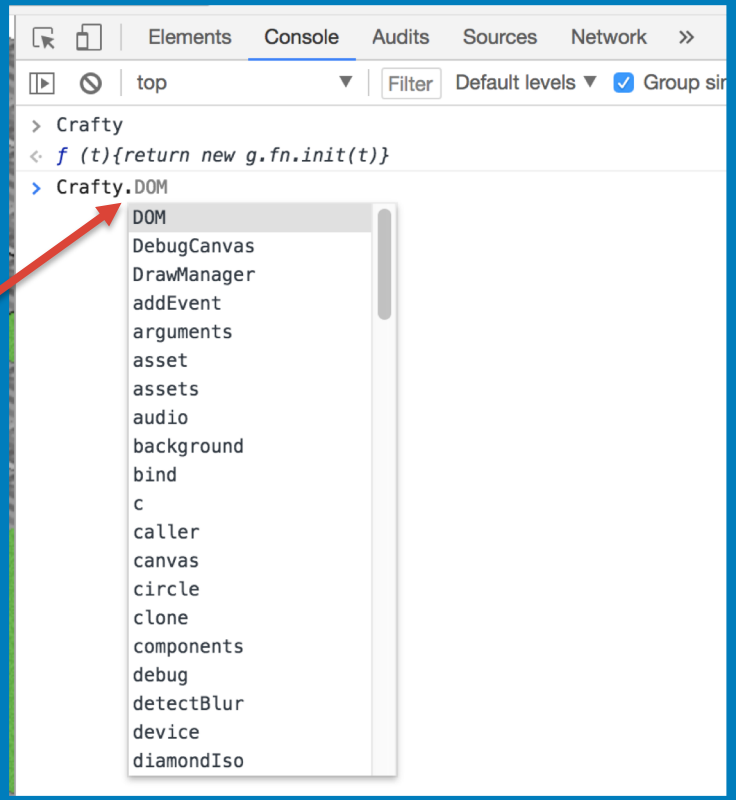
#FFFD38

#6F359E



Introducing the ‘Crafty’ Object

- Crafty JS is a Javascript game library (engine) that helps us to build games in Javascript
- To use it we my add the crafty.min.js file to our HTML page with a <script> tag
- Type ‘Crafty’ in Chrome Dev Tools, console tab and then . (dot)

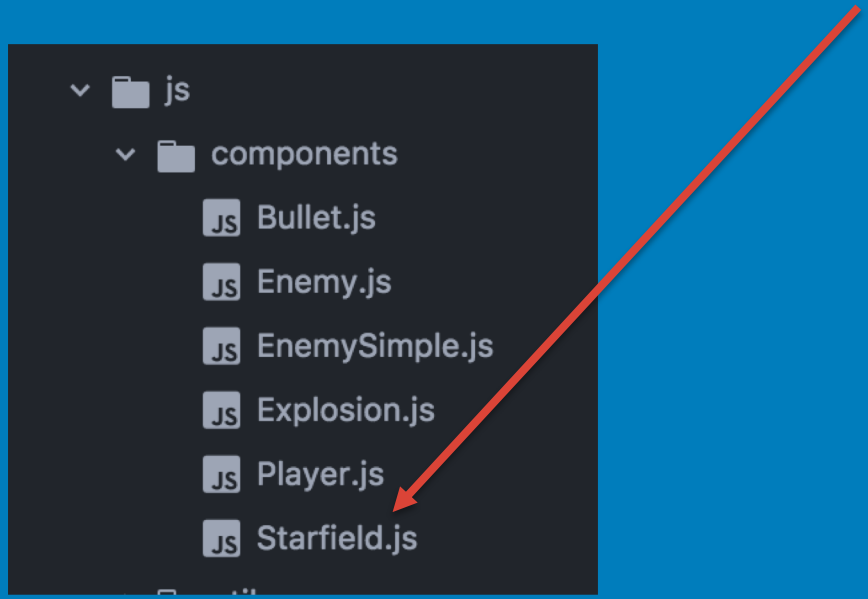


NEXGEN CODECAMP

Lesson 2 – Creating a Starfield

Creating a Starfield Object

The Starfield code is found in js/components/Starfield.js



We use the ‘new’ keyword to create new Javascript Objects:

```
// Create the Starfield
__starField = new Starfield(game);
```

Once, created, `__starField` can now be used to create a moving star Field in the game. You can have Different numbers of stars running at different speeds

The Starfield function takes 1 parameter (game)

```
// Start the __starField
__starField.start();
```

To call a method on an object, we use the dot notation

Looking at the code for the Starfield in Starfield.js, we can see that we could alter the velocity, size and number of stars quite easily.

```
function Starfield(div) {
  this.fps = 30;
  this.canvas = null;
  this.width = 0;
  this.width = 0;
  this.minVelocity = 5;
  this.maxVelocity = 40;
  this.stars = 200;
  this.maxSize = 2;
  this.intervalId = 0;

  this.initialise(div);
}
```

NEXGEN CODECAMP

Lesson 3 – Adding a Player

NEXGEN CODECAMP

What is a Sprite ?

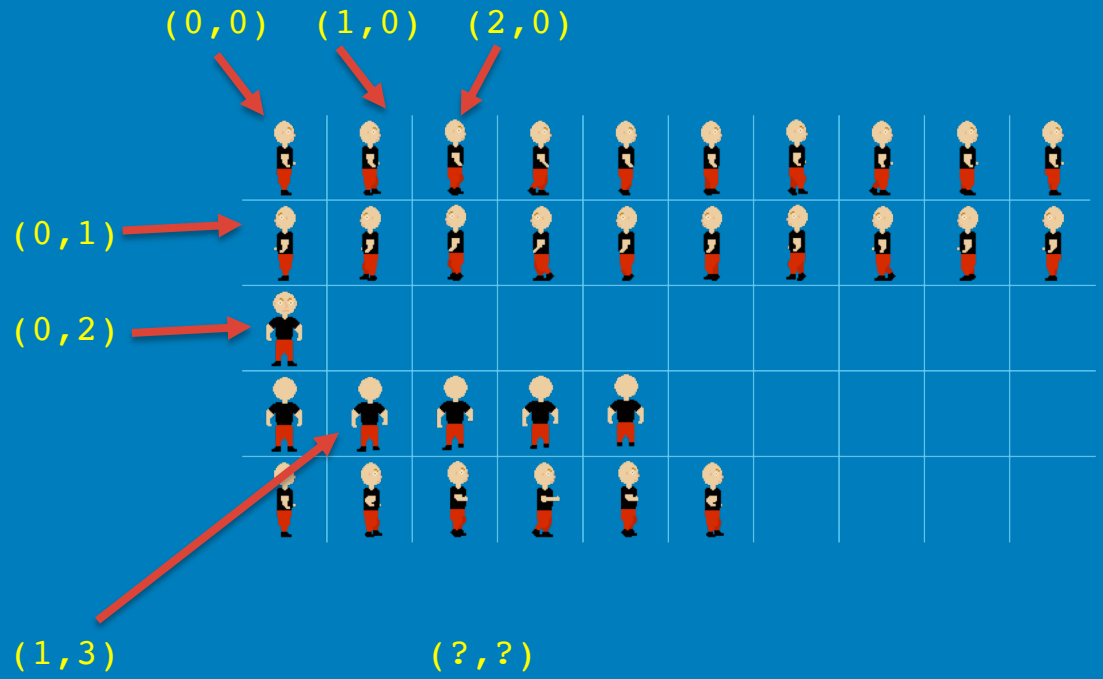
sprite

/spraɪt/

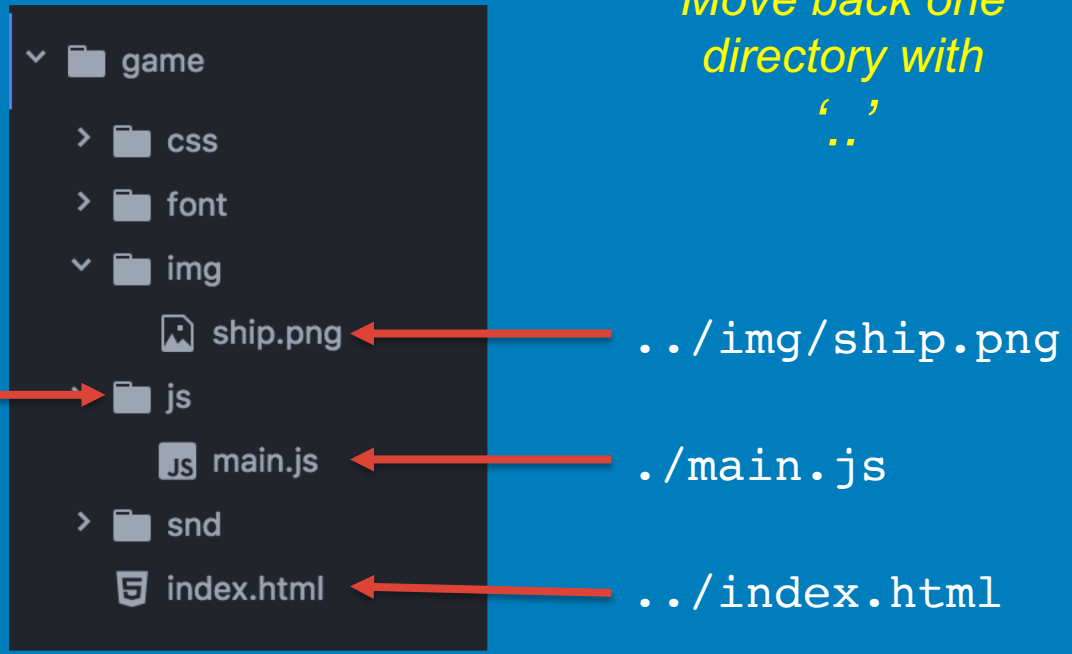
noun

1. an elf or fairy.
   synonyms: fairy, elf, pixie, imp, brownie, puck, peri, goblin, hobgoblin, kelpie, leprechaun; More
2. a computer graphic which may be moved on-screen and otherwise manipulated as a single entity.

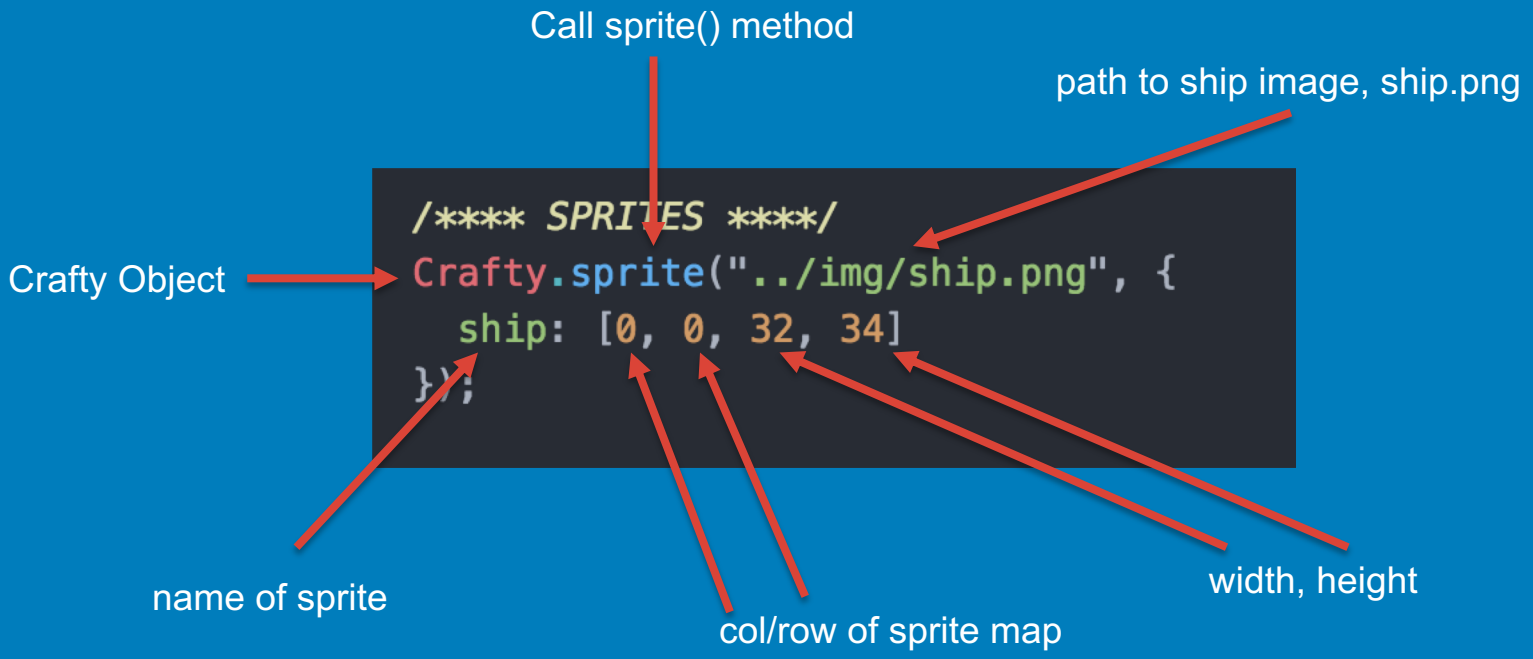
Sprites on a Grid



Relative Directories



Create a Sprite with Crafty



NEXGEN CODECAMP

Entities in Crafty

An entity is some *thing* in your game. If that sounds vague, it's because practically anything can be an entity -- the player, the enemies, a projectile, a high score counter, or a menu item.

Entities are made up of components, which you can think of as bundles of functionality. Crafty provides several built-in components, and you can also define your own with Crafty.c().

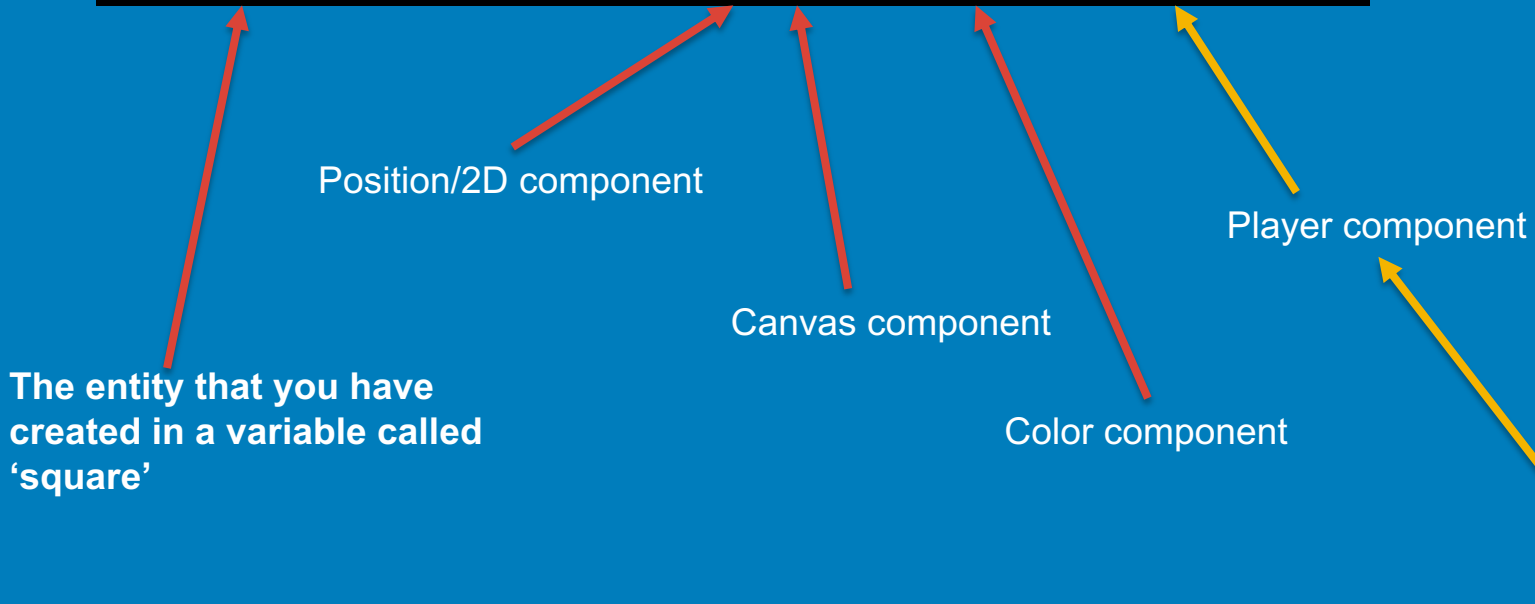
Creating an Entity:

You create an entity with Crafty.e(). In most cases you'll pass in a list of the components you want to start with:

```
var square = Crafty.e("2D, Canvas, Color");
```

Components in Crafty

```
var __player = Crafty.e("2D, Canvas, Color, Player");
```



Player.js file

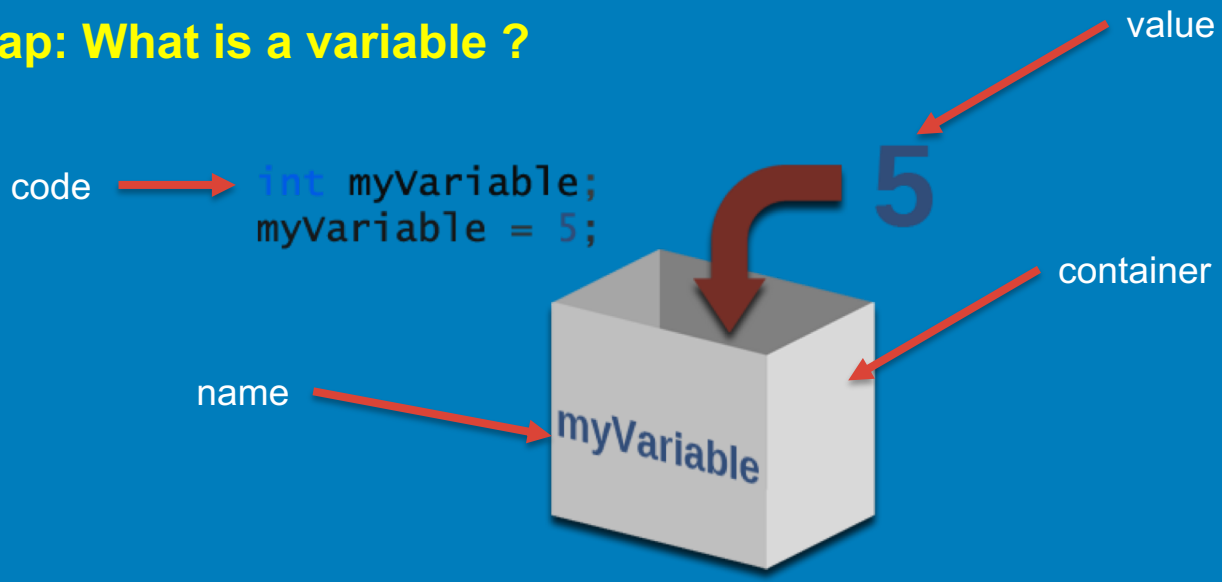
```
// Create Player Component
Crafty.c("Player", {
  required: "2D, Canvas, Twoway, SpriteAnimation, Collision, ship",
  init: function() {
    // Put initialisation code, eg. position
  },

  bindEvents: function(that) {
    that.bind("KeyDown", function(e) {
      // Respond to a Key pressed down
    }).bind("EnterFrame", function(e) {
      // Do something every frame of animation
    }).bind('EnemyHit', function(data){
      // Do something when you have hit an Enemy, eg. add to score
    }).onHit('EnemySimple', function(o) {
      // Do something when player is hit, eg. lose a life
    });
  },

  fire: function() {
    // Fire a bullet code!
  },

  talk: function() {
    // Write to console when the component is ready
    Crafty.log("Player 1 Ready!")
  }
})
```

NEXGEN CODECAMP

Lesson 4 – Variables & an Enemy

Variable Scope

Recap: What is a variable ?



Variable Scope

```
function testVariableScope(){
    var myVar = "I am here!";
}
```

Create a variable inside a function in Chrome Dev tools.

Type the variable name and then ENTER

```
> myVar
▶Uncaught ReferenceError: VM732:1
myVar is not defined
    at <anonymous>:1:1
>
```

What just happened and Why?

Variable Scope

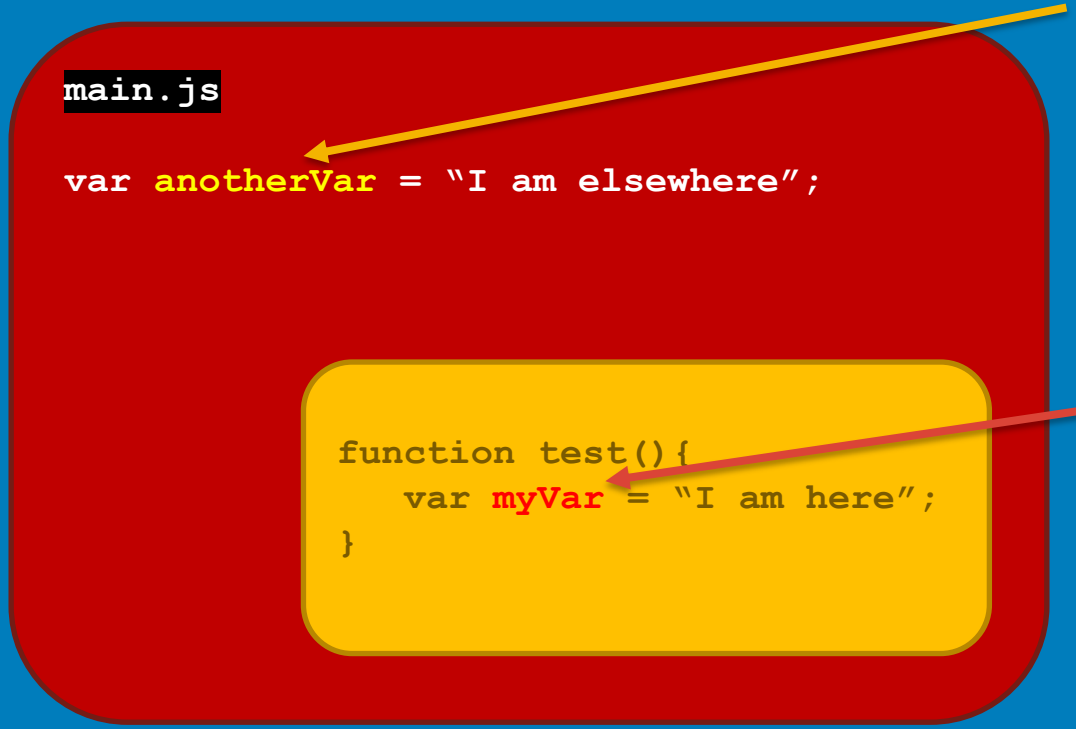
```
> myVar = "I am here!"
<· "I am here!"
>
```

This time just type out the variable but not in a function

Again, type the variable on the Chrome Dev tools command line and press ENTER

```
> myVar
<· "I am here!"
>
```

Variable Scope



The **scope** of this variable is *global*. It can be accessed from anywhere inside the program.

The **scope** of this variable is *local* to the function. That is, it can only be accessed from code that sits inside the function.

NEXGEN CODECAMP

Lesson 5 – Adding Component Behaviour

Create a Component

Create a Crafty component entity

```
Crafty.c("EnemySimple", {
  required: "2D, Canvas, SpriteAnimation, Collision, Gravity  bee",
  /* This function will be called when the component is added to an entity */
  init: function() {…

  afterInit: function(props){…

  bindEvents: function(self) {…

  talk: function() {…

  fire: function() {…
})
```

Give the component Gravity

Components give you power!

```
afterInit: function(props){
  this.x = props.x;
  this.y = props.y;
  this.reel('EnemySimpleMove', 1000, [[0, 0],[1, 0]]);
  this.animate('EnemySimpleMove', -1);
  this.gravity();
  this.gravityConst(props.speed);
},
```

Adding the Gravity component, gives your component/entity extra functions. You can now set the gravity speed.

NEXGEN CODECAMP

Lesson 6 – Multiple enemies!

Let's say I had a character thus:

...and a function `function step()`

... that makes the character take **one** step.

... What if you wanted him to take 10 steps?

You could call the function 10 times...?

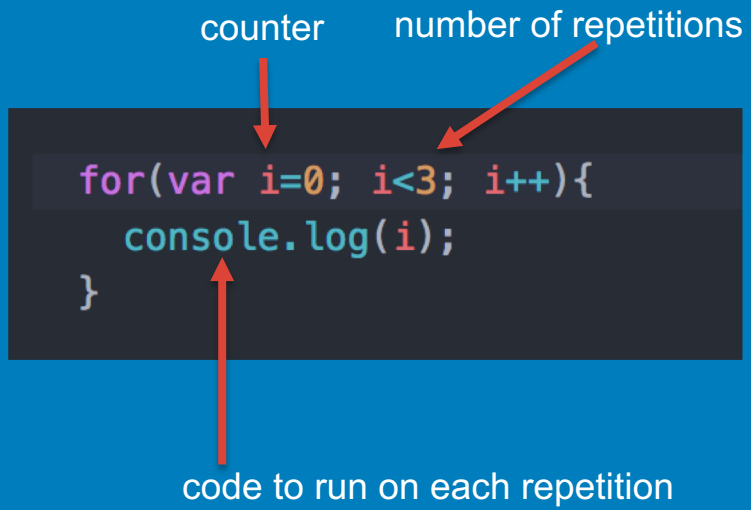
```
step()
step()
step()
step()
step()
step()
step()
step()
step()
step()
```

Is that a good idea?

What if you wanted to take 1000 steps?

For....Loops

Another way of repeating an instruction!



NEXGEN CODECAMP

Lesson 7 – Events

Events in a Game

Level complete

Enemy is killed

Player loses a life

Game started

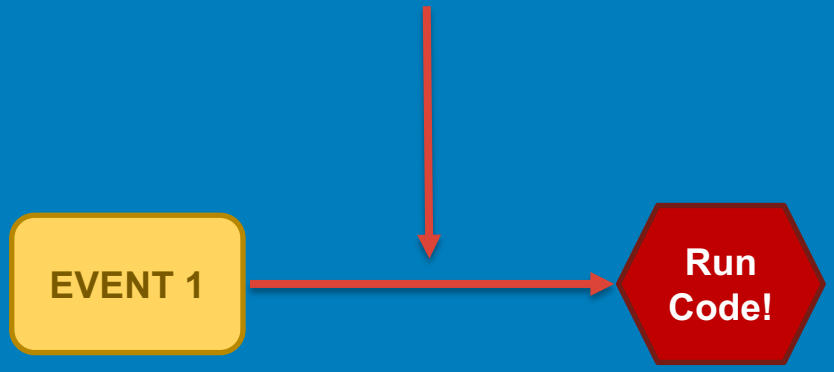
End of game

Score updated

High score reached

Events in a Game

When an EVENT occurs, we *may* want to run some Javascript code!



Events in a Game

To **setup** an event in Crafty, we must '**bind**' it

Events in a Game

To **run** an event in Crafty, we must **trigger** it

Data that is passed into the ’player_fired’ event code