

# WELCOME TO INTRO TO JAVASCRIPT

Are you ready for class? Please make sure you have:

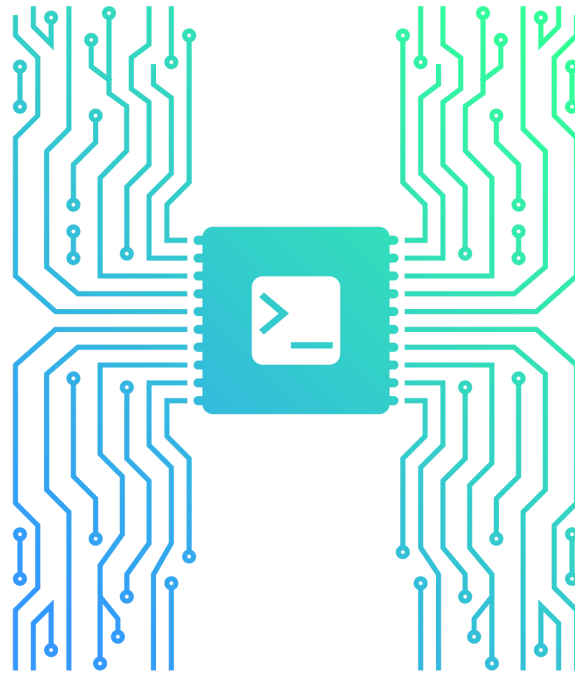
- **A text editor:** Atom is a good open source option - <https://atom.io/>
- **A modern web browser:** Chrome or Firefox are solid options.

# INTRO TO JAVASCRIPT

This class is presented by Women in Coding in partnership with the Fayetteville Free Library.



Thanks for Sponsoring!



HACK  
UP STATE

# WELCOME!

Some quick "rules":

- Every question is important.
- Help each other.
- Have fun.

## THIS MORNING:

- Quick intro
- What is JavaScript
- Setup & Dev tools
- JavaScript & Code Challenges

# WELCOME!

Tell us about yourself.

- Who are you?
- What do you hope to get out of the class?
- What is your favorite ice cream flavor?



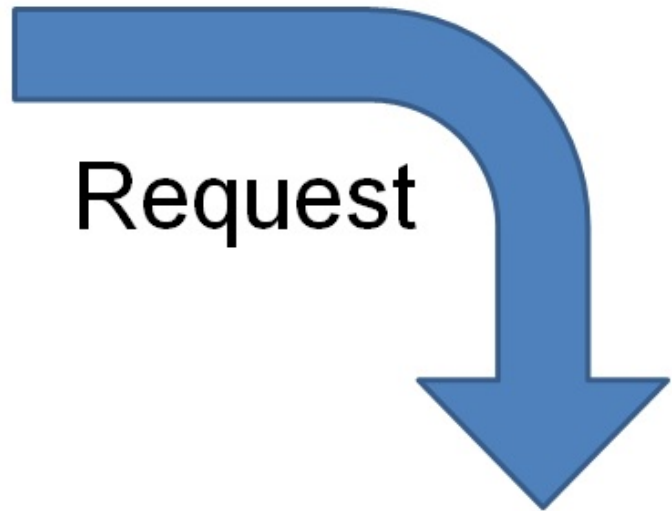
# WHAT IS JAVASCRIPT?

JavaScript is the language of the web



Client

Request



Server

Response

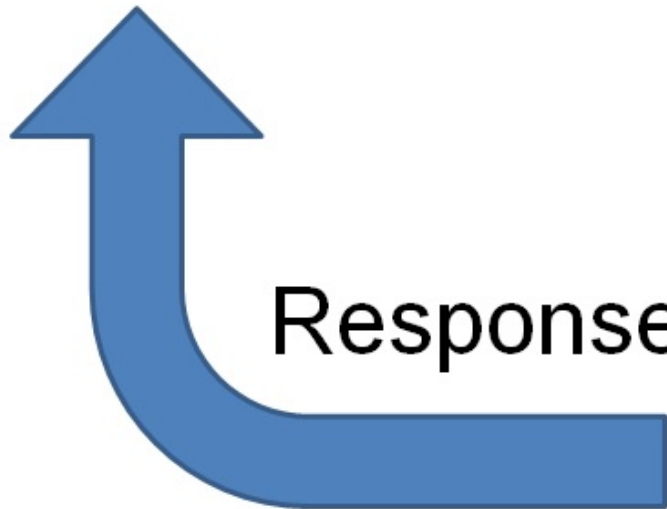


Photo credits: [Andrew E. Larson](#) and [John Seb Barber](#) cc

# WHAT IS JAVASCRIPT?

- Created by Brendan Eich as "LiveScript" in 1995, which got renamed to "JavaScript"
- Standardized by the **ECMAScript** specifications. This class covers **ES5** (standardized in 2009)
- A **client-side processing language**. A browser reads the code and runs it directly.
- Interfaces with **HTML & CSS**.
- Lets you build dynamic webpages that respond to input from users.

# WHAT IS JAVASCRIPT?



JavaScript is not Java

# WHAT CAN JAVASCRIPT DO?

## Image lightboxes

Note: ID's are single use and are only applied to one element.  
Galleries are created from elements who have the same "data-fancybox-group" or "rel" attribute value.



Script uses the `href` or `data-fancybox-href` attribute of the matched elements to obtain the location of the content and to figure out content type you want to display. You can specify type directly by adding classname (fancybox.image, fancybox.iframe, etc) or `data-fancybox-type` attribute. Use `title` or `data-fancybox-title` attribute to specify item caption.

Various types	HTML	JavaScript
<ul style="list-style-type: none"><li>• <a href="#">Ajax</a></li><li>• <a href="#">Iframe</a></li><li>• <a href="#">Inline</a></li><li>• <a href="#">SWF</a></li></ul>		<ul style="list-style-type: none"><li>• <a href="#">Youtube (iframe)</a></li><li>• <a href="#">Google maps (iframe)</a></li><li>• <a href="#">Non-existing url</a></li></ul>

Alternatively, you can set content type as an option: `$( ".open_ajax" ).fancybox({ type: 'ajax' });`.

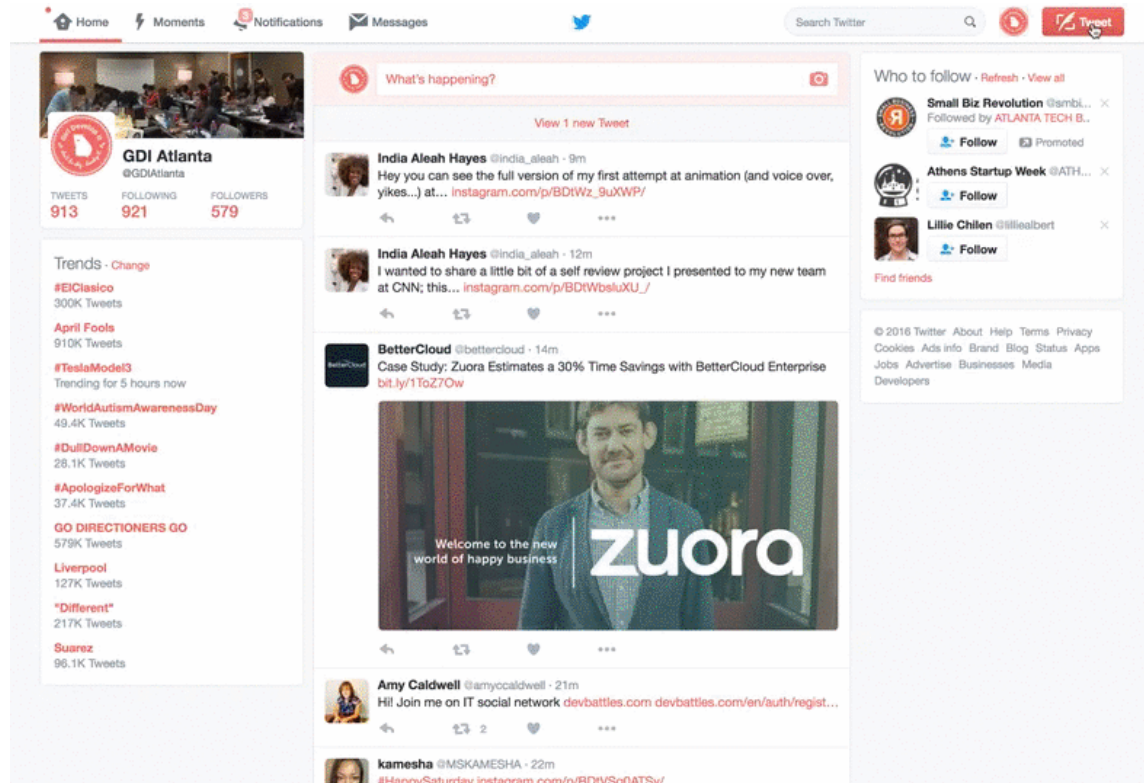
Note, ajax requests are subject to the [same origin policy](#). If fancyBox will not be able to get content type, it will try to guess based on 'href' and will quit silently if would not succeed (this is different from previous versions where 'ajax' was used as default type or an error message was displayed).

### Extended functionality

**Remember to include the necessary files!** Each helper is located in separate files.

# WHAT CAN JAVASCRIPT DO?

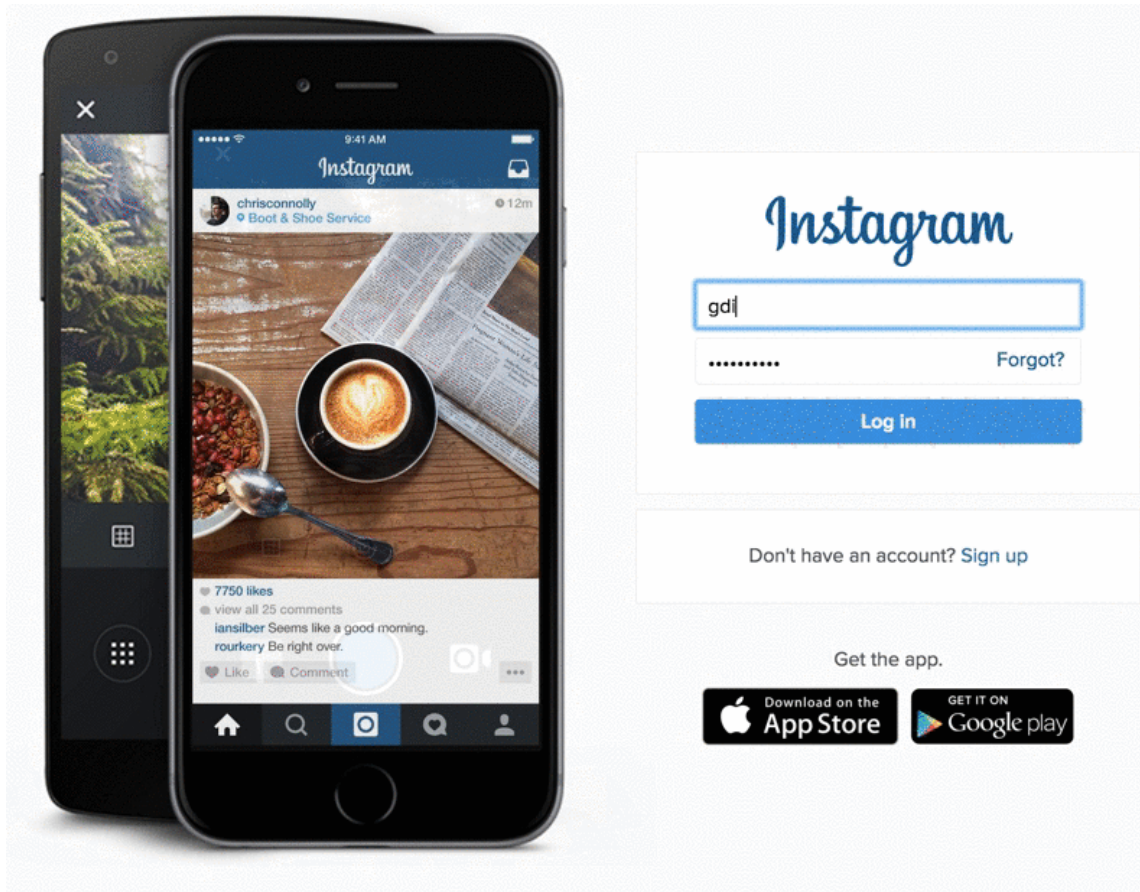
Fully featured web applications





# WHAT CAN JAVASCRIPT DO?

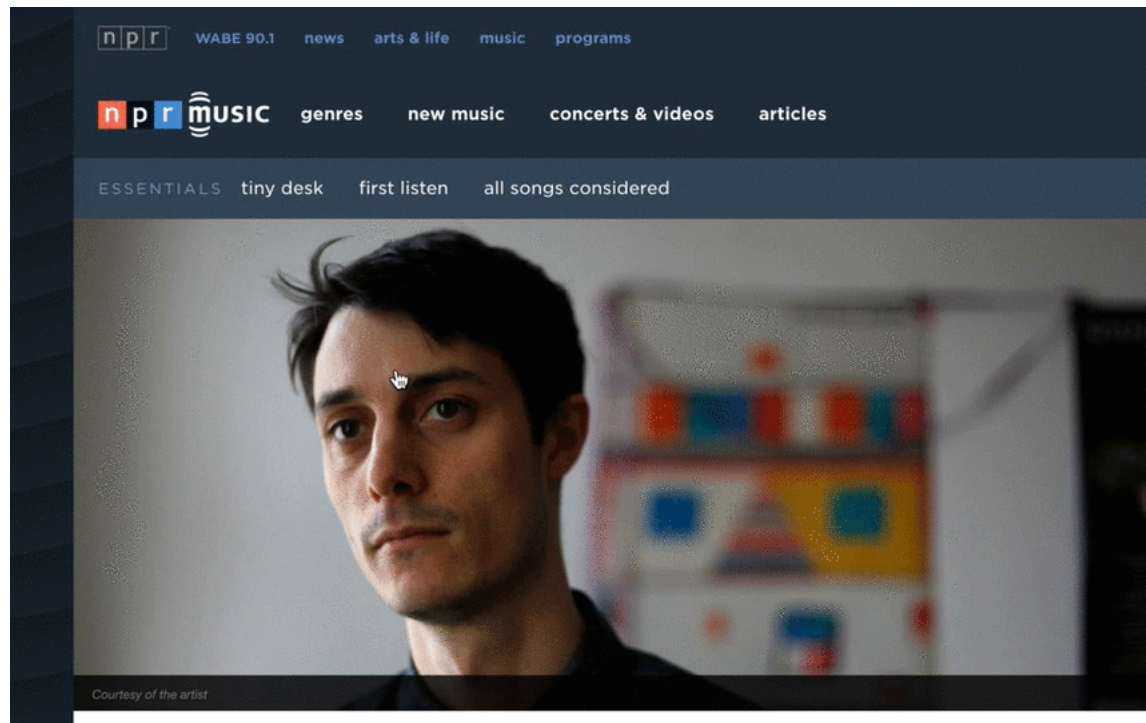
Keep track of users with [Cookies](#) or storing data with [local storage](#).





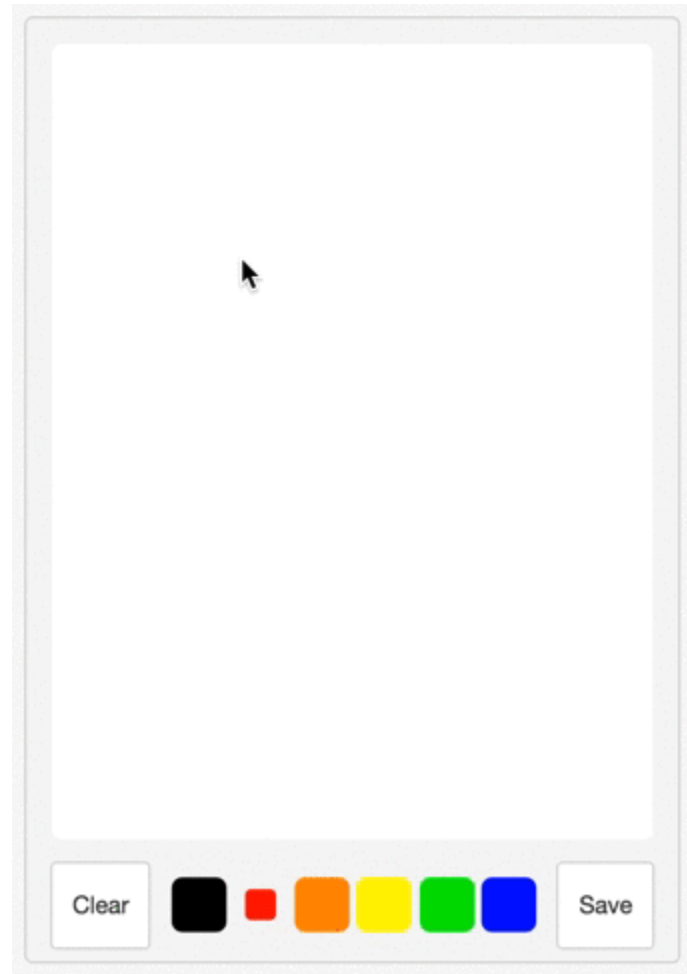
# WHAT CAN JAVASCRIPT DO?

Interactive elements like tabs, sliders, etc



# WHAT CAN JAVASCRIPT DO?

Drawing & animation



# SETUP

Let's get setup to write your first JavaScript program. Make a folder called "learn-js", and inside make a new file called "index.html". Place this code inside.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Test Page</title>
  </head>
  <body>
    <p>This is my JavaScript testing page.</p>

    <script>

    </script>
  </body>
</html>
```

# INCLUDING JAVASCRIPT IN HTML: SCRIPT TAGS

You can mix JavaScript and HTML. The script tag tells your browser the stuff inside is code, not content.

```
<script>  
    CODE GOES HERE  
</script>
```

# JAVASCRIPT FILES

Just like CSS, you can split JavaScript out into its own file and link to it into your HTML.

```
<script src="path/to/file.js"></script>
```

# THINKING LIKE A PROGRAMMER

Computers are great at *processing*, but they are bad at *understanding*.

When you write a program, you must break down every step into simple pieces.



# HOW DOES JAVASCRIPT WORK?

1. You visit a website with JavaScript code on it.
2. Your browser (e.g., Chrome) reads the code line-by-line.
3. The browser runs each line of code as it reads it.
4. Based on these instructions, the browser performs calculations and changes the HTML and CSS on the page.
5. If the browser finds code it doesn't understand, it stops running and creates an error message.

# CONSOLE

You can see what's going on in the **console**.





Elements

Console

Sources

Network



Timeline

Profiles

Resources

Security

Audits





top ▼ ☐ Preserve log

> console.log('hello, panel!');  
hello, panel!

VM194:1

< undefined

>

# SEPARATING INSTRUCTIONS

JavaScript is written in statements, each statement must end with a semicolon.

```
<script>  
  alert('Hello World!');  
  console.log('I am glad to meet you');  
</script>
```

# COMMENTS

Comments are notes in your code that people can read and computers will ignore.

```
<script>  
  /*I can wrap long comments  
  with multiple lines  
  like this*/  
  console.log('Hello World!'); //Or mark short comments like this  
</script>
```

# MAKING TEXT APPEAR ONTO YOUR SCREEN

Open an alert box.

```
alert('Hello World!');
```

Display a message in your console.

```
console.log('Hello World!');
```

Add something to the page.

```
document.write('Hello World!');
```

## LET'S DEVELOP IT

- Open your "index.html" file.
- Add a comment to the code.
- Try different ways of making a message appear on screen.
  - `console.log('your message');`
  - `alert('your message');`
  - `document.write('your message');`
- Create a new file called `mycode.js`. Move your code to this file and link it to your page.

# VARIABLES

A variable is a place to store a value.

## VARIABLE VALUES

- When you first create a variable, it does not have a value (it is undefined).
- You can set a value for a variable.
- Variables can hold different types of information, like words, numbers, and collections of data.
- The value of a variable can be changed.

# NAMING VARIABLES

- Variable names are case-sensitive.
- A new variable should have a unique name.
- Variable names need to start with a letter, \$, or \_.
- Avoid reserved words.
- Choose clarity and meaning for humans to read later.



# DECLARING A VARIABLE

To declare (create) a variable, just type the word "var" and the variable name.

```
<script>  
  var numberOfKittens;  
</script>
```

It is a good idea to give your variable a starting value. This is called initializing the variable.

```
<script>  
  var numberOfKittens = 5;  
</script>
```

## USING A VARIABLE

Once you have created a variable, you can use it in your code. Just type the name of the variable.

```
<script>  
  var numberOfKittens = 5;  
  console.log (numberOfKittens);  
</script>
```

## LET'S DEVELOP IT

In your JS file, create a variable and give it a valid name and a value. Then, display the value.

# DATA TYPES

- **string** string of characters

```
var userName = 'Jane Lane';
```

- **number** integer or floating point

```
var myAge = 30;
```

- **boolean** true or false

```
var catsAreBest = true;
```

- **undefined** value that hasn't been defined

```
var favoriteThings;
```

- **null** an explicitly empty value

```
var goodPickupLines = null;
```

# NUMBERS

Variables can hold numbers, either integers or floats (decimals).

```
<script>  
  var numberOfKittens = 5;  
  var cutenessRating = 9.6;  
</script>
```

JavaScript automatically converts integers to floats

NaN = Not-A-Number

# ARITHMETIC OPERATORS

Once you have numbers, you can do math with them!

```
<script>  
  var numberOfKittens = 5;  
  var numberOfPuppies = 4;  
  var numberOfAnimals = numberOfKittens + numberOfPuppies;  
</script>
```

# ARITHMETIC OPERATORS

Example	Name	Result
$-a$	Negation	Opposite of a.
$a + b$	Addition	Sum of a and b.
$a - b$	Subtraction	Difference of a and b.
$a * b$	Multiplication	Product of a and b.
$a / b$	Division	Quotient of a and b.
$a \% b$	Modulus	Remainder of a divided by b.

## LET'S DEVELOP IT

Create two variables and try some arithmetic operators.  
Don't forget to display your results!



# STRINGS

Variables can also hold strings.

A string is a group of characters wrapped in single, or double, quotes.

```
<script>  
  var kittensName = 'Fluffy';  
</script>
```

If you want to use a quote in your string, you'll need to "escape" it with a backslash.

```
<script>  
  console.log('I\'d like to use an apostrophe');  
</script>
```

# STRING OPERATORS

**Concatenation:** you can put strings together with a plus sign (the concatenation operator).

```
<script>
  var kittensName = 'Fluffy ';
  var fullName = kittensName + ' McDougale';
  console.log(fullName); //Outputs 'Fluffy McDougale'
</script>
```

# STRING OPERATORS

You can also use += to add things to the end of a string.

```
<script>
  var kittensName = 'Admiral ';
  kittensName += ' Snuggles';
  console.log(kittensName); //Outputs 'Admiral Snuggles'
</script>
```

## LET'S DEVELOP IT

Create two variables, a first name and a last name, and then put them together to make a full name. Don't forget to display your results!

## QUICK RECAP: CODE SEARCH

In this code, spot the comment, variables, and operators.

```
var billPreTip = 10;
var tipPercent = 0.15; // Can be changed

var billTip = billPreTip * tipPercent;
var receipt = 'Meal: ' + billPreTip + ' Tip: ' + billTip +
' Total: ' + (billPreTip + billTip);

console.log(receipt);
```

# **FUNCTIONS**

Functions are separable, reusable pieces of code.

# USING FUNCTIONS

First, declare the function.

```
<script>
  function turtleFact() {
    console.log('A turtle\'s lower shell is called a plastron. ');
  }
</script>
```

Then, use it as many times as you want!

```
<script>
  turtleFact();
</script>
```

# ARGUMENTS

Functions can accept input values, called arguments.

```
<script>
  function callKitten (kittenName){
    console.log('Come here, ' + kittenName + '!');
  }
  callKitten ('Fluffy'); //outputs 'Come here, Fluffy!'

  function addNumbers(a, b) {
    console.log(a + b);
  }
  addNumbers(5,7); //outputs 12
  addNumbers(9,12); //outputs 21
</script>
```



# ARGUMENTS

You can also pass variables into functions. These variables do not need to have the same name as the function arguments.

```
<script>
  function addOne(inputNumber){
    var newNumber = inputNumber + 1;
    console.log('<p>You now have ' + newNumber);
  }
  //Declare variables
  var numberOfKittens = 5;
  var numberOfPuppies = 4;
  //Use them in functions
  addOne(numberOfKittens);
  addOne(numberOfPuppies);
</script>
```

## LET'S DEVELOP IT

Turn the code you wrote to output someone's full name into a function that accepts the arguments `firstname` and `lastname`, then use it.

# RETURNING VALUES

You can have a function give you back a value, to use later.

```
<script>
  function square(num) {
    return num * num;
  }
  console.log(square(4)); // outputs '16'.
  var squareOfFive = square(5); // will make squareOfFive equal 25.
</script>
```

Return will immediately end a function.

## LET'S DEVELOP IT

Add a return statement to your name function. Use that function to set the value of a variable.

# BOOLEAN VARIABLES

Boolean variables represent the logical values True and False

```
var catsAreBest = true;  
var dogsRule = false;
```

If you try to use another variable as a boolean, JavaScript will guess.  
0, "", undefined, null are FALSE, everything else is TRUE

# THE IF STATEMENT

Use if to decide which lines of code to execute, based on a condition.

```
if (condition) {  
  // statements to execute  
}
```

```
var bananas = 5;  
if (bananas > 0) {  
  console.log ('You have some bananas!');  
}
```

# COMPARISON OPERATORS

Example	Name	Result
<code>a == b</code>	Equal	<b>TRUE</b> if a is equal to b (can be different types).
<code>a === b</code>	Identical	<b>TRUE</b> if a is equal to b, and the same type.
<code>a != b</code>	Not equal	<b>TRUE</b> if a is not equal to b (can be different types).
<code>a !== b</code>	Not identical	<b>TRUE</b> if a is not equal to b, or they are not the same type.
<code>a &lt; b</code>	Less than	<b>TRUE</b> if a is strictly less than b.
<code>a &gt; b</code>	Greater than	<b>TRUE</b> if a is strictly greater than b.
<code>a &lt;= b</code>	Less than or equal to	<b>TRUE</b> if a is less than or equal to b.
<code>a &gt;= b</code>	Greater than or equal to	<b>TRUE</b> if a is greater than or equal to b.

# **WATCH OUT!**

Don't mix up =, ==, and ===



## LET'S DEVELOP IT

Make a variable called "temperature." Write some code that tells you to put on a coat if it is below 50 degrees.

# THE IF/ELSE STATEMENT

Use else to provide an alternate set of instructions.

```
var age = 28;
if (age >= 16) {
  console.log ('Yay, you can drive!');
} else {
  console.log ('Sorry, but you have ' + (16 - age) + ' years until
}
```

# THE IF/ELSE IF/ELSE STATEMENT

If you have multiple conditions, you can use else if.

```
var age = 20;
if (age >= 35) {
    console.log('You can vote AND hold any place in government!');
} else if (age >= 25) {
    console.log('You can vote AND run for the Senate!');
} else if (age >= 18) {
    console.log('You can vote!');
} else {
    console.log('You can\'t vote, but you can still write your repres
```

## LET'S DEVELOP IT

Modify your "wear a coat" code for these conditions:

1. If it is less than 50 degrees, wear a coat.
2. If it is less than 30 degrees, wear a coat and a hat.
3. If it is less than 0 degrees, stay inside.
4. Otherwise, wear whatever you want.

# THERE IS SO MUCH MORE!

Keep learning:

- Basic JavaScript on [Free Code Camp](#).
- JavaScript at [Code Academy](#).
- [Eloquent JavaScript](#) Digital Book with Exercise sandbox
- Wes Bos' [JavaScript 30](#) is a 30 day coding challenge with 30 JavaScript projects for you to complete.

Additional Resources:

- [Internet Fundamentals](#) from Front End Masters is a great video course overview of how the internet works.
- [JavaScript Guide](#), from the Mozilla Developers Network.
- Thanks to [Girl Develop It](#) for making their teaching material available.