INTRO TO JAVASCRIPT

Thanks to Women in Coding for hosting and organizing this event!

Women in Coding
Providing affordable coding classes for women in the community

Thanks for Sponsoring!

StartFast

Code

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 children's book?

WHAT IS JAVASCRIPT?

JavaScript is a client-side programming language.

That means a browser reads the code and runs it directly.

WHAT DOES JAVASCRIPT DO?

JavaScript interfaces with HTML and CSS to create dynamic webpages that respond to input from users.

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>
```

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 a popup 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 index.html.
- 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

- The variable name is case-sensitive.
- A new variable needs to have a unique name.
- Variable names need to start with a letter, $, or _.
- Variable names can only be made of letters, numbers, $, or _.

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.

NUMBERS

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

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

The browser will automatically convert integers to floats if needed

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 + ' McDougle';
    console.log(fullName); //Outputs 'Fluffy McDougle'
</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!

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, 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
a == b	Equal	**TRUE** if a is equal to b (can be different types).
a === b	Identical	**TRUE** if a is equal to b, and the same type.
a != b	Not equal	**TRUE** if a is not equal to b (can be different types).
a !== b	Not identical	**TRUE** if a is not equal to b, or they are not the same type.
a < b	Less than	**TRUE** if a is strictly less than b.
a > b	Greater than	**TRUE** if a is strictly greater than b.
a <= b	Less than or equal to	**TRUE** if a is less than or equal to b.
a >= b	Greater than or equal to	**TRUE** 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.

KEEP LEARNING:

- **Check out StartFast Code!**
- Basic JavaScript on Free Code Camp for interactive JavaScript lessions.

Additional Resources:

- JavaScript Guide, from the Mozilla Developers Network.
- Thanks to Girl Develop It for making their teaching material available.