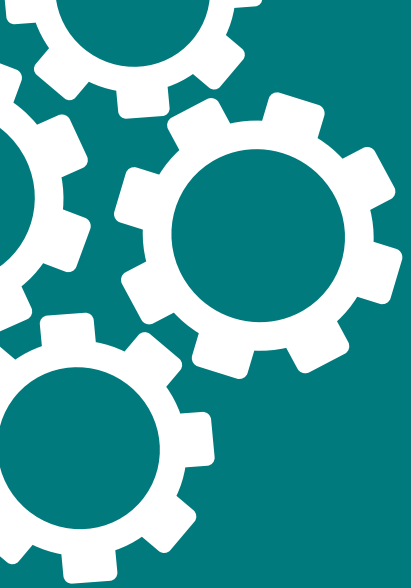

Introduction to Javascript

WOMEN WHO CODE /london

00

Before we start

console.log

Allows us to print information

```
console.log("Hello World");

Output: Hello World
```

We use console.log to print information in the parentheses.
This is useful for us if we want to log/debug information or in general to see the value of the data we've stored.
We will be using console.log throughout this presentation and in postman

01

Variable

What is a variable?

A variable is a way to store data and use it later on by labelling it

```
const x = 3;
const name = "Parisa";
```

1) **Declare** variable - let, const, var
let - create variable, reassign value
const - can't reassign value
var - declared on the topmost scope

2) **Variable name** - combination letters, numbers, currency sign/underscore, can use single or double quotes. No spaces

3) **Assignment** =

4) **Value**

5) **Semi colon**

Variable Scope

Global and Private Scope

```
let firstName = "Jake";

{
let firstName = "Parisa";
console.log(firstName); //output Parisa
}

console.log(firstName); //output Jake
```

First log will print Parisa (Private Scope)
Second log will print Jake (Global Scope)

Variables defined in the code block are available inside the code block.
Variables defined outside of code block are available outside and inside code block

02

Data Types

Different Data Types

String - letters, :), use single or double quotes
Booleans - true/false
Numbers - numeric values

undefined - default value assigned to variable when you create it. There is no value and you don't assign initial value to it
null - clearing our value

03

Arrays

Intro to Arrays

A data structure is a way to store multiple values in the same variable
We can store: strings, numbers, booleans, dates, arrays in arrays
Can access individual values starting from 0

```
const array = ["a","b","c"];

console.log(array[0]); Output: a
console.log(array[1]); Output: b
console.log(array[2]); Output: c
console.log(array[3]); Output: undefined
```

Intro to Arrays II

```
1)Add element to an array
array.push("d");
Output: array = ["a", "b", "c", "d"]

2)Delete element from an array
array.pop("d");
Output: array = ["a", "b", “c"]

3)Add element to start of an array
array.unshift("d");
Output: array = ["d", "a", "b", “c"]
```

04

Functions

Intro to Functions

Blocks of code that make them easily repeatable, thus reducing errors

```
function sum(a,b) {
 return a + b;
}

console.log(sum(1,3));
```

1) **function** key word
2) **Function name** - combination letters, numbers, currency sign/underscore
3) **Parameters**
4) **Logic in curly braces**

Call function by using function name and adding parentheses in brackets

Anonymous and Arrow functions

Anonymous - same rules as a function in terms of having an input, output and logic but we don't give them a name.
This is an example of a callback function - input to another function and executes asynchronously.
We're calling the anonymous functions after 1 second.

```
setTimeout(function () {
   console.log("Hey");
},1000);
```

Arrow Functions - has one line of code only, we can drop curly braces and drop the semi colon and there's no need to add a return value here:

```
setTimeout(() => console.log("Hey"), 1000)
```

05

Objects

Objects are data types that describe things in detail and have properties

```
1)Declare Object
let x = {};

2) Add properties to an object:
x = {
name:"Parisa",
isOlderThan21: true,
"+30": true};

3)Can change the property value:
x.name = "Nora";

2)Access property of an object:
x["+30"];
Output: true
```

Object methods - perform an action

```
const person = {
name: "Parisa",
isOlderThan21: true,
"+30": true,
sayHello: function(){return 'Hello, my name is ' +
this.name;}
};

console.log(person.sayHello());
Output: Hello Parisa
```

Convert object to JSON

```
const person = {
name: "Parisa",
isOlderThan21: true,
"+30": true,
sayHello: function(){return 'Hello, my name is ' + this.name;}
};

let json = JSON.stringify(person);

1) console.log(person)
   Output: { name: 'Parisa',
             isOlderThan21: true,
             '+30': true,
             sayHello: '[Function: sayHello]' }
2)console.log(json)
Output: {"name":"Parisa","isOlderThan21":true,"+30":true}
```

06

Classes

Classes

What if we want to create multiple objects with the same properties but different values.
Template to create objects to provide details so all objects have the same structure.

To create a class use the keyword class and then the class name (class name starts with capital)
Constructor are rules to create the object and support properties: name, isOlderThan21
Constructor instantiates the object to avoid repeated code

```
class Object {
constructor(name, isOlderThan21) {
this.name = name;
this.isolderthan21 = isolderthan21;
}
}

const myObject = new Object("Parisa", true);
```

Add function to object and class

```
const x = {
name: "Parisa",
isOlderThan21: true,
"+30": true,
eatMeat: function() {
console.log(“I am eating beef”);
}
};

Method - function belonging to a class.
Here we don’t need function keyword

eatMeat() {
console.log("I am eating beef");
}
```

07

Conditionals

Conditionals

If else statements - certain action if the condition is true or another action otherwise

The parentheses after if store a boolean condition

You can use || (or), && (and) for conditions

```
if(booleanExpression) {
//Branch if true
} else {
//branch if false
}
```

Example

We can declare variables we will use in our conditional at the start (taking into account variable scope).
Then we add if, else (sometimes if, else if (if we have multiple conditions).

```
let lights = "red";
let go = false;

if(lights == "green") {
  go = true;
}
else {
  go = false;
}
```

08

Loops

For Loop

Allow us to repeat code a set number of times

```
const names = ["James","Alex","Jane","Claire"];

for(let i=0; i < names.length; i++){
console.log(names[i]);
}
//Output James Alex Jane Claire
```

1) **Initialise**
2) **Count - less than the array length**
3) **Increment - Add one**
4) **Logic inside curly braces will be executed**

While Loop

Repeat until they give you correct information

```
let password = "tomorrow";
let userGuess = "";

while(userGuess != password) {
 userGuess = prompt("Please enter your
password");
}
```

1) **While keyword to start**
2) **Parentheses**
3) **Increment - Add one**
4) **Logic inside curly braces will be executed**

Do While Loop Example

```
let i = 0;
const n = 5;

do {
    console.log(i);
    i++;
}
while(i < n);

What will print out?
This will print: 0,1,2,3,4
```

Rule of thumb

1) **Do while loop**

- **When you need code to run once and check condition after**

2) **While loop**

- **Check condition before it runs code**
- **Condition to meet before we stop the loop**

3) **For loop**

- **How many times we want to run the loop**

08

Postman

Libraries

1) Chai.js library

i) Expect variable in response to be equal to certain value

ii) Expect a 200 response

iii) Check content-type

iv) Check response type

2) Moment.js

i) Date time stamps

ii) Comparing dates with another date

iii) var today = moment()

3) Faker.js

i) Creating data on the fly

ii) $random - cats, name, organisation

Some examples

```
Example 1
var jsonData = pm.response.json();
pm.collectionVariables.set("email".jsonData.response.email);


Example 2
let email = pm.collectionVariables.get("email");
pm.expect(email).to.equal("parisa@example.com");


Example 3
pm.test("Status code is 200", function() {
  pm.response.to.have.status(200);
});

Example 4
let email = pm.collectionVariables.get("email");
console.log(email)
console.warning(email)
console.info(email)
console.error(email)
```

Helpful tip

Code snippets are on the right hand side.

POST https://postman-echo.com/

Send

Params Authorization Headers Body Pre-request Tests Settings Cookies

```
pm.test(`Returns correct email: ${pm.collectionVariables.get('email')}`, function () {
    //add your expectation here
    let email = pm.collectionVariables.get('email');
    pm.expect(email).to.equal("dawn.ellis@example.com");
})


var jsonData = pm.response.json();
pm.collectionVariables.set("variable_key", "variable_value");
console.log(jsonData.data.email);
```

Test scripts are written in JavaScript, and are run after the response is received. Learn more about tests scripts ↗

Snippets

- Set a global variable
- Set a collection variable
- Clear an environment variable
- Clear a global variable
- Clear a collection variable
- Send a request
- Status code: Code is 200
- Response body: Contains string

Helpful links

https://dev.to/sethusenthil/var-vs-let-vs-const-1cgc
https://testautomationu.applitools.com/javascript-tutorial/
https://web.stanford.edu/class/cs103/tools/truth-table-tool/
https://www.postman.com
https://www.chaijs.com
https://momentjs.com/docs/
https://www.youtube.com/watch?v=juuhb3W8xT4&list=PL6iUkDSEH9SsXETWu9Jhg2mmOfu3TU05Q

Thank you

Follow us!
https://linktr.ee/wwcode_london

WOMEN WHO CODE®
/london