What is JavaScript?

- A language for the web browser *and* the server (node.js)
- High level (very abstracted), interpretted programming language (not pre-compiled, more or less true)
- Implements the ECMAScript® Language Specification
- Is multi-paradigm (you can write your code in many different ways)

Why learn JavaScript? 

- You can build lots of things with it!
- Iteractive user interfaces using awesome frameworks
    - angular

    - vue
    - react
- Very fast server side applications
    - express
- Mobile apps
    - react native
    - NativeScript
- Even desktop applications
    - electron

What will we learn?

- This is a crash course.
- Just the basics, no frameworks (yet)
- `variables` and `data types`
- `conditionals`
- `arrays`
- `objects`
- `loops`
- `functions`
- Keep on learning!
    - mdn
    - codecademy
    - udemy
    - many more

How do we use JavaScript?

index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>How do we use JavaScript?</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <!--we want our HTML and CSS to load first-
    <script src="script.js"></script>
  </body>
</html>
```

script.js

```
console.log('hello world!');
```

Variables

```
let age = 30; // can be reassigned
age = 31; // yes
const name = 'Radu'; // can NOT be reassigned
name = 'Tudor'; // nope
```

```
TypeError: Assignment to constant variable.
```

```
const name; // MUST be initialized
```

```
SyntaxError: Missing initializer in const declaration.
```

> Always use `const` unless you're sure you want to reassign that value.

Data types

- numbers

```
10
-50
3.14159
```

- strings

```
'hello world'
"double quotes are also cool"
```

- booleans

```
true
false
```

- undefined and null

```
undefined
null
```

- object

```
{
  name: 'Claudia',
  age: 29
  isLoggedIn: true
}

// arrays are also objects
typeof [1, 2, 3] // object
typeof null // object 🤦‍♂️
/*
Because the spec says so.
This is generally regarded as a mistake.
*/
```

autoboxing

```
const name = 'Radu';
typeof name; // string
name.length; // 4
```

> This a process called autoboxing. When you call a property on a primitive types, JavaScript will first convert it into a temporary wrapper object, and access the property / method on it, without affecting the original.

String concatenation

normal addition

```
const name = 'Radu';
const age = 30;
const message = 'My name is ' + name + ' and I
// My name is Radu and I am 30;
```

template strings

```
const name = 'Radu';
const age = 30;
const message = `My name is ${name} and I am ${
// My name is Radu and I am 30;
// We MUST use backticks!
```

Conditionals

if

```
const myAge = 30;
const yourAge = 30;

if (myAge > yourAge) {
  console.log('I am older than you.');
}
```

else

```
const myAge = 30;
const yourAge = 30;

if (myAge > yourAge) {
  console.log('I am older than you.');
}
else {
  console.log('I am younger than you.');
}
```

else if

```
const myAge = 30;
const yourAge = 25;

if (myAge > yourAge) {
  console.log('I am older than you.');
}
else if (myAge === yourAge) {
  console.log('We are the same age.');
}
else {
  console.log('I am younger than you.');
}
```

Checking for equality

==

- Will compare for equality after doing any necessary type conversions

```
const myAge = 30;
const yourAge = '30'

myAge == yourAge; // true
```

===

- Will *not* do the conversion

```
const myAge = 30;
const yourAge = '30'

myAge === yourAge; // false
```

> There are many things to say here but most importantly, always use `===` .

Arrays

- Lists that hold multiple values
- Are homogeneous (can store mixed data types)
- We can access individual elements or add new ones.

```
const names = ['Silvia', 'Victor', 'Robert'];
const values = ['test', 10, true, null, [1, 2]]
```

declaration

> Remember that arrays are zero based. The first element has index `0` .

```
const name = names[0]; // Silvia
```

index

```
const name = names[0]; // Silvia
```

length

```
console.log(names.length); // 3
```

```
.push()
```

```
names.push('Radu');
console.log(names); // 'Silvia', 'Victor', 'Rob
console.log(names.length); // 4
```

> There are more array methods, read about them here.

> While an array is of type `object`, you can check if a value is an array using `Array.isArray()`.

The for loop

- Loops offer a quick and easy way to do something repeatedly.
- There are many different kinds of loops, but they all essentially do the same thing.
- They repeat an action some number of times.

```
structure
```

```
for ([initialExpression]; [condition]; [increme
  statement
```

```
implementation
```

```
for (let i = 0; i < 3; i++) {
        console.log(i)
}
// 0
// 1
// 2
```

arrays

```
const names = ['Silvia', 'Victor', 'Robert'];

for (let i = 0; i < names.length; i++) {
        console.log(names[i])
}
// 'Silvia'
// 'Victor'
// 'Robert'
```

> Always be careful to not write an infinite loop.

```
for (let i = 0; i < 3; i--) {
        console.log(names[i])
}
// 0
// -1
// -2
// Infinity...
```

Object literals

- a collection of related data

- key : value pairs
- used when storing information
- makes stuff easier to read
- useful sending / receiving a request to / from the server

declaring properties

```
const person = {
  name: 'Silvia',
  age: 21,
  isLoggedIn: true
};

person.name; // Silvia
person.age; // 21
```

reading properties

```
person.name; // Silvia
person.age; // 21
person.height; // undefined
```

writing properties

```
person.address = 'Park Street no. 11';
```

Functions

- One of the fundamental building blocks in JavaScript
- A function is a JavaScript procedure—a set of statements that performs a task or calculates a value.
- To use a function, you must define it somewhere in the scope from which you wish to call it.

declaration

- name
- list of parameters (optional)
- statements enclosed in curly brackets `{}`

```
function sayHello() {
  console.log('hello world');
}
```

invocation

```
sayHello();
// hello world
```

one parameter

```
function square(number) {
  return number * number;
}
const number = square(4); // 16
```

multiple parameters

```
function add(a, b) {
  return a + b;
}
const sum = add(100, 300); // 400
```

Thanks!