

INTRODUCTION TO JAVASCRIPT

Summary

- JS is syntactically similar to C++
- Is interpreted
 - Node.js
 - Web console
- Supports dynamic typing and OOP



INDEX

1. Values, Types, and Operators
2. Program Structure
3. Functions
4. Data Structures: Objects and Arrays



1. Values, Types, and Operators

Values

- Strings
 - Single/Double Quotes
 - “Hello World” is the same as ‘Hello World’
 - Backticks or Template Literals
 - Allow definition of multiline strings without escaping and allow to use **variables** inside the string



1. Values, Types, and Operators

Values

- Undefined values
 - Null
 - Undefined
 - ''
 - []
 - {}
 - ...



1. Values, Types, and Operators

Types

- *typeof* operator
 - Types List:
 - Undefined
 - Boolean
 - Number
 - String
 - BigInt
 - Symbol
 - Null
 - Object
 - Function



1. Values, Types, and Operators

Types

- Dynamic Typing
 - `x = 1`
`x = '1'`
- Implicit Conversions
 - `1 + '1'`



1. Values, Types, and Operators

Operators

- `===` VS `==`
 - `===` doesn't implicitly do type conversion when comparing



1. Values, Types, and Operators

Operators

- `//`
 - `x = null || "Hello World!"`
- `&&`
 - `x = true && "Bye bye!"`



2. Program Structure

let vs var

- **const** the variable can not be reassigned, but its content can.
- **let** declare variables in local scope
- **var** declare variables in global scope
 - It is not recommended to use var



2. Program Structure

Environment Functions

```
'use strict';
```

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

```
prompt('Enter your name:');
```

```
alert('Your password is too short!');
```

```
console.log(Math.max([1, 4, 7, 2, 3]));
```

```
console.log(Math.cos(0.53));
```

```
console.log(Math.floor(9.9999));
```



2. Program Structure

Conditionals and loops

```
'use strict';
```

```
const a = 10;
if (a === 10) {
  console.log('if');
}
switch (a) {
  case 10:
    console.log('switch');
    break;
  default:
    console.log('Error');
}
```

```
'use strict';
```

```
let i = 0;
while (i < 10) {
  console.log(i);
  i++;
}
```

```
i = 0;
do {
  console.log(i);
  i++;
} while (i < 10);
```

```
for (let i = 0; i < 10; i++) {
  console.log(i);
}
```



3. Functions

Type of Functions

- Bindings
 - `const` sample = `function`(`args`) { ... }
- Declared
 - `function` sample(`args`) { ... }
 - These can be used anywhere in the block they are on
- Arrow
 - `const` sample = (`args`) => { ... }



3. Functions

Arguments

- If you pass more than the required args JS will ignore it.
- If you pass less than the required args Js will put *undefined* in them
- Also has default value for args.



4. Data Structures: Objects and Arrays

Arrays

- Properties
 - .length
 - .toUpperCase() | .toLowerCase()
 - .pop() | .push()
 - ...



4. Data Structures: Objects and Arrays

Arrays

- Spreading
 - Given an array you can extend this array with another array.
 - Ej:
 - `let a = [1, 2, 3];`
`let b = [...a, 4, 5, 6];`



4. Data Structures: Objects and Arrays

Arrays

- Destructuring
 - Given an array you can assign the elements of this array with the elements of another array.
 - Ej:
 - `let a = [1, 2, 3];`
`let [b, c, d] = a;`



4. Data Structures: Objects and Arrays

Arrays

- Loops
 - for (const element of list) { ... }
 - element will be each element of the list.



4. Data Structures: Objects and Arrays

Objects

- Are the native synonymous of Hash Tables in C++ (pairs **Key/Value**).
 - const object = {
 key1: value1,
 key2: value2,
 ...
}



4. Data Structures: Objects and Arrays

Objects

- Assignment:
 - `Object.assign(a, b);` // assigns copies of the keys of object b in a, if there are equal keys, will be overwritten.



4. Data Structures: Objects and Arrays

Objects

- Loop:
 - `for (const element in object) { ... }`
 - element will be each key of object.



4. Data Structures: Objects and Arrays

Objects

- Methods:
 - An object can contain functions that will be treated like methods.
 - Also can contain other objects, arrays, etc



4. Data Structures: Objects and Arrays

Objects

- Implicit Keys:
 - An object can contain objects like keys and his value will be the object's value.



4. Data Structures: Objects and Arrays

Objects

- JSON:
 - Are files that contains a JS object.
 - Useful to save and load big amount of data into an object.



4. Data Structures: Objects and Arrays

Objects

- In:
 - True if key is in the Object, False otherwise.



Conclusion

- Javascript is really similar to c++ in its syntax
- It is weakly-typed, unlike c++ which is strongly-typed
- It is dynamically typed, letting variable represents any value
- Objects are its base since they allow to express data and functionalities in a very flexible way



Questions

- If you have questions, this is your time



Bibliography

- *Eloquent JavaScript*
 - Theme 1-4
- *The Modern JavaScript Tutorial*
 - Part 1



