JavaScript

JavaScript is a multiparadigm programming language. It follows a blend of several programming paradigms.

- At its core, JavaScript is a functional programming language. However, over the years, cahnges have been added to expand it.
- So it has capabilities to satisfy other paradigms (like OOP).
- JavaScript is mainly used as a ClientSide Language (Front-End Developement).
- It can be capable of running both client and server side (typically to use JS server side, you use a technology like node.js)

Some key points...

- JS is a loosely typed language. (No need => `String word;`)
  - we don't need to declare the type of a variable
  - Variables don't need to adhere to a single type, once they're created. Variables can change types when they are assigned a new value.
- Flexibility over Structure
- JIT (Just in Time Complication)
  - the code is actually compiled (right before) as it runs.

`Java`	`JavaScript`
`- compiled language`	`- scripting language`
`- strong, static typing`	`- weak, dynamic typing`
`- OOP`	`- Functional (multiparadigm)`
`- Server-side`	`- Client-side`
`- Classes`	`- No classes (can simulate)`
`- Classical Inheritance`	`- Prototypal Inheritance`
`- Access Modifiers`	`- No Access Modifiers (Define scopes)`
`- semi-colon required`	`- optional semi-colon`

`Ham`	`Hamster`

Data Types of JavaScript

- JS has primitives and Objects
- variables are not declared a type, but instead are of a certain type, based on values assigned to them.

Primitives:

- number
- boolean
- string
- null
- undefined

Objects:

- array
- functions
- many more (user-defined Objects, Date, etc.)

null: indicates that a variable has no value **undefined**: indicated that the value of a variable is unknown.

JS is pass-by-value for primitives and pass-by-reference for Objects.

Objects: are collections of key-value pairs.

- can be created with constructors and the `new` keyword
- generally we create object literals

```
myObject = {
    prop1: value1,
    prop2: value2
};
```