# JavaScript

“JavaScript is the only language that I’m aware of that people feel they don’t need to learn before they start using it.”

-DOUGLAS CROCKFORD

# Overview

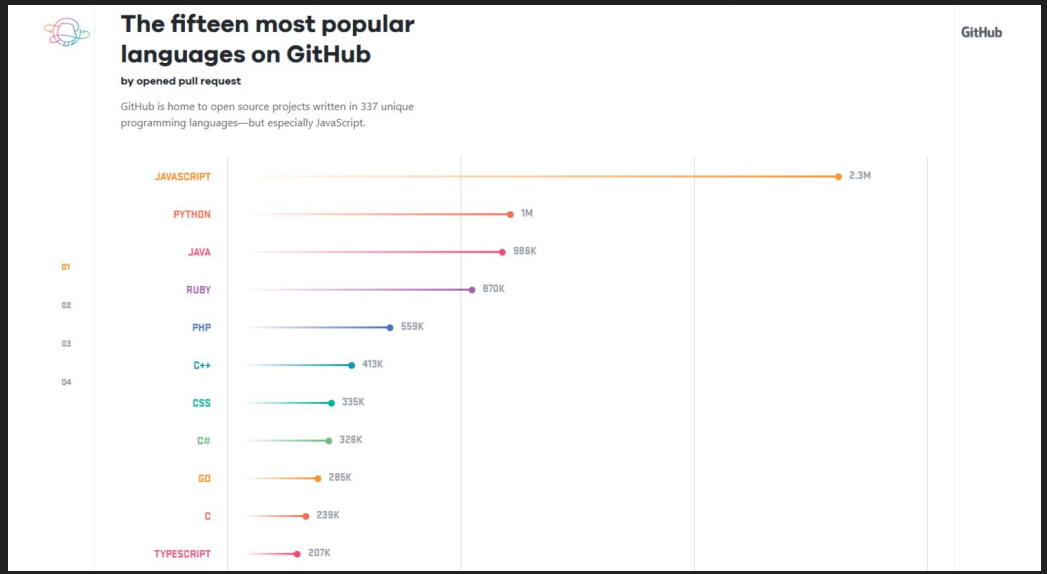
1. **Course Admin**
2. **The Language**
3. **Basics of Web Dev + Why Use JS**
4. **basic intro to setting up a play env. + node**
5. **Example (variables and types)**
6. **Example (functions)**
7. **Example (Objects)**
8. **Example (This and scoping)**
9. **Example (error handling)**

# Course

- Split up into 3 main parts
  - Basics of the Language
    - Why js
    - Basic Examples
    - How to get started
  - The Dom Tree and You
    - What is the DOM tree?
    - How do events work within the DOM
  - Asynchronicity
    - Ye Olde Way
    - Promises
    - Ajax Principle
- Labs/Tutes
  - We will release a repo of exercises and code soon

# JavaScript

- Why teach it?
- It’s popular
- It’s useful
- It’s easy to pick up and hence poorly understood
- very easy to pick up because of it’s “everything is a object” system
- Makes reacting to events easy



# How does it run?

- Interpreted just in time compilation.
- No single implementation, many implementations of the ECMA spec
  a. Rhino (Mozilla)
  b. V8 (Chrome, Opera, Node.JS)
  c. JavaScriptCore (Safari)
  d. Chakra (Edge)
  e. etc

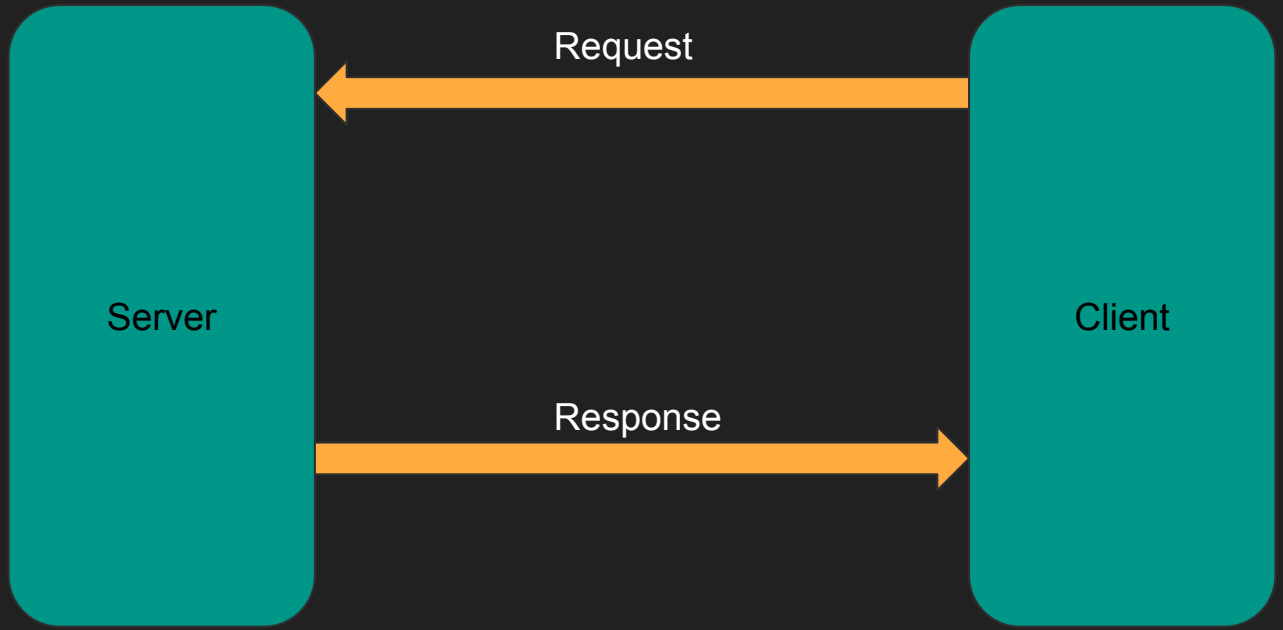
# Disclaimer

JavaScript has a lot of things that it *can* do but that you *shouldn't* do. And various things that the language is trying to shift away from and things it's trying to move towards.

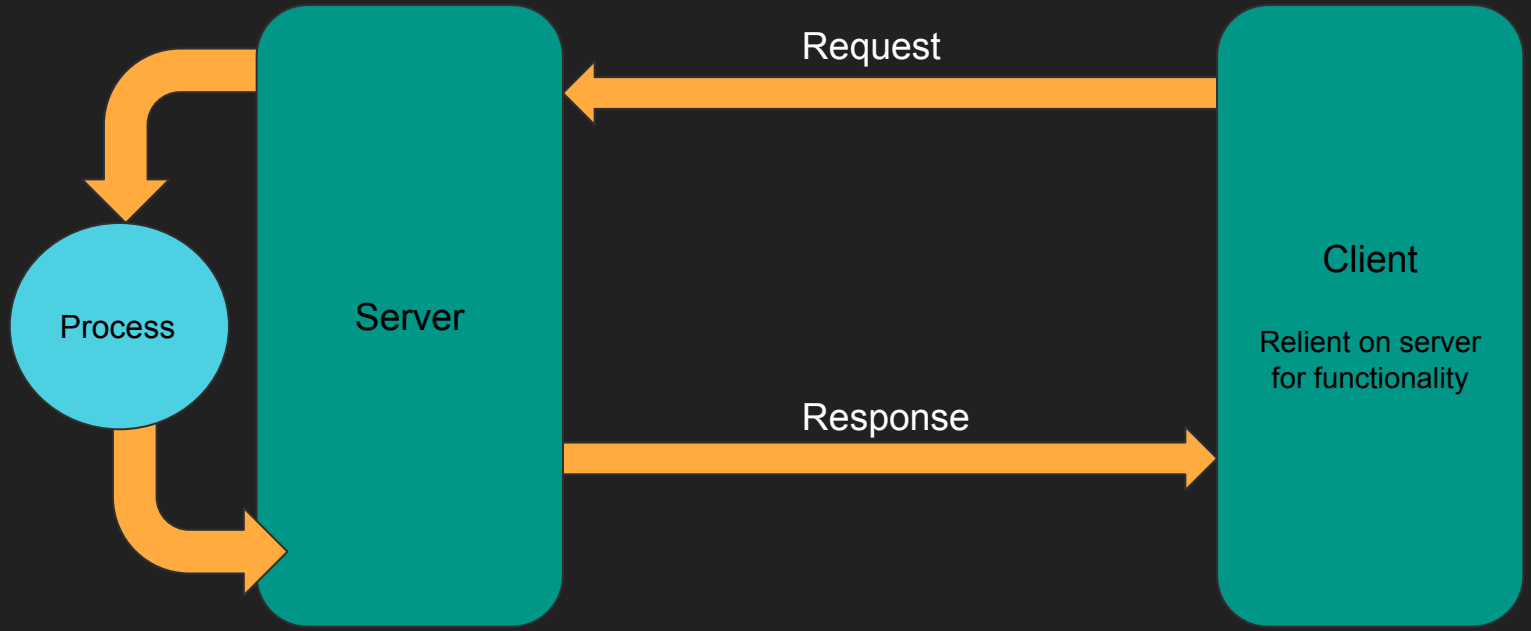
It's the simple fact that js was originally designed for hacking together simple interactivity for pages and then had to get patched over time to become a full programming language without breaking legacy code.

# Ye Olde Web Dev

[Example 1]

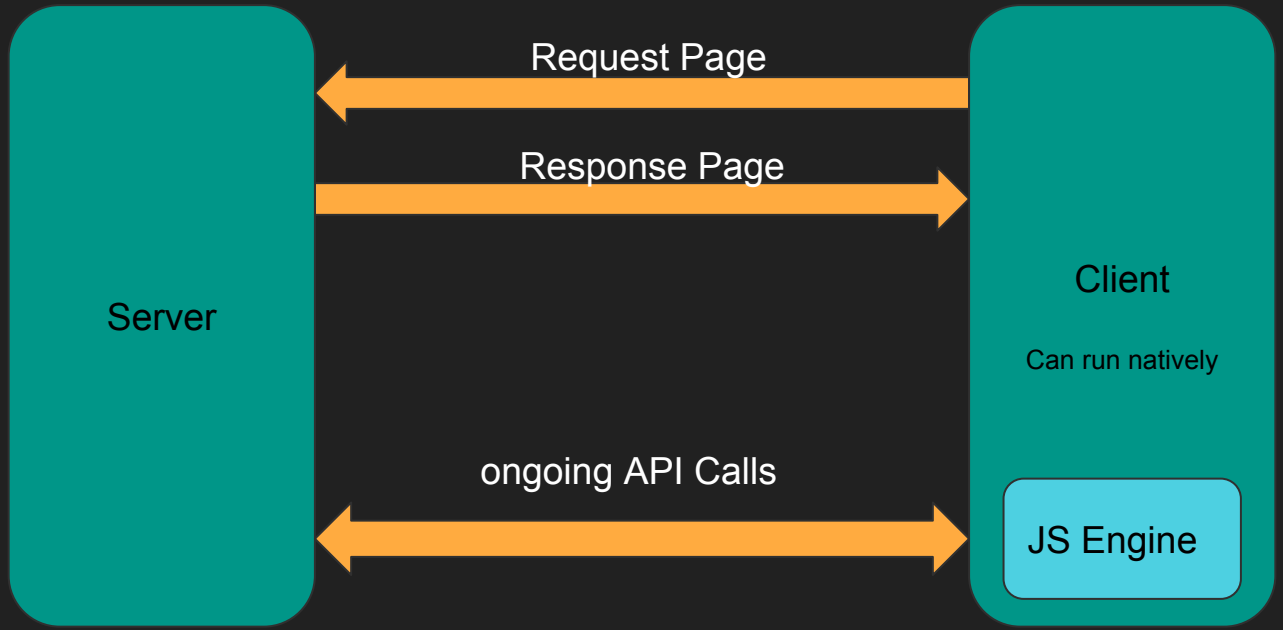


# Ye (less) Olde Web Dev



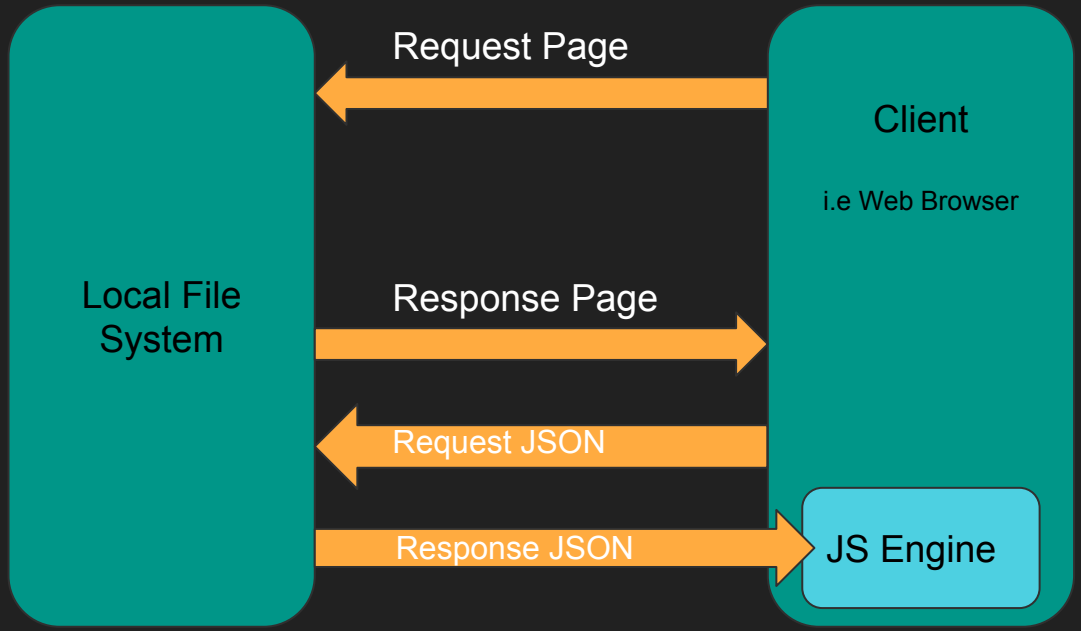
# Ye Newe Web Dev

[Example 2]



# Playing Around

[Example 3]

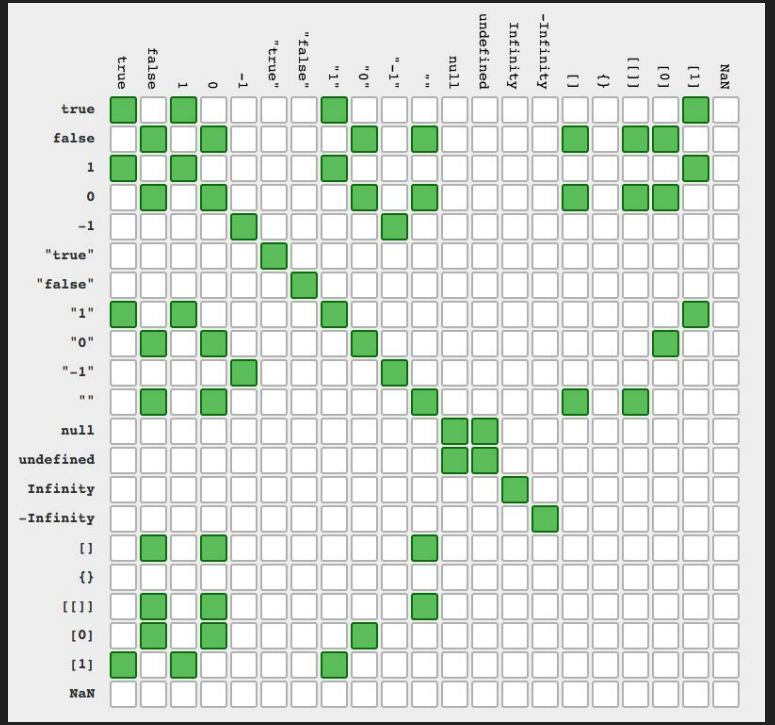



JS Engine

# Variables And Types

[Example 4]

- let vs const vs var vs nothing
- arrays
- for loops
- type coercion



# Functions

[Example 4]

- Parameters
- Scope
- Hoisting (w/ functions)

# Objects

[Example 5]

- Making a new object
- filtering / accessing
- nesting + functions
- ES6 classes (old + new)
- game

# This and Scoping

[Example 6]

- This
- Function functions
- anon functions
- all cloned
  - prototype model
  - how prototypes work

# Error Handling

[Example 6]

- try catch