JavaScript

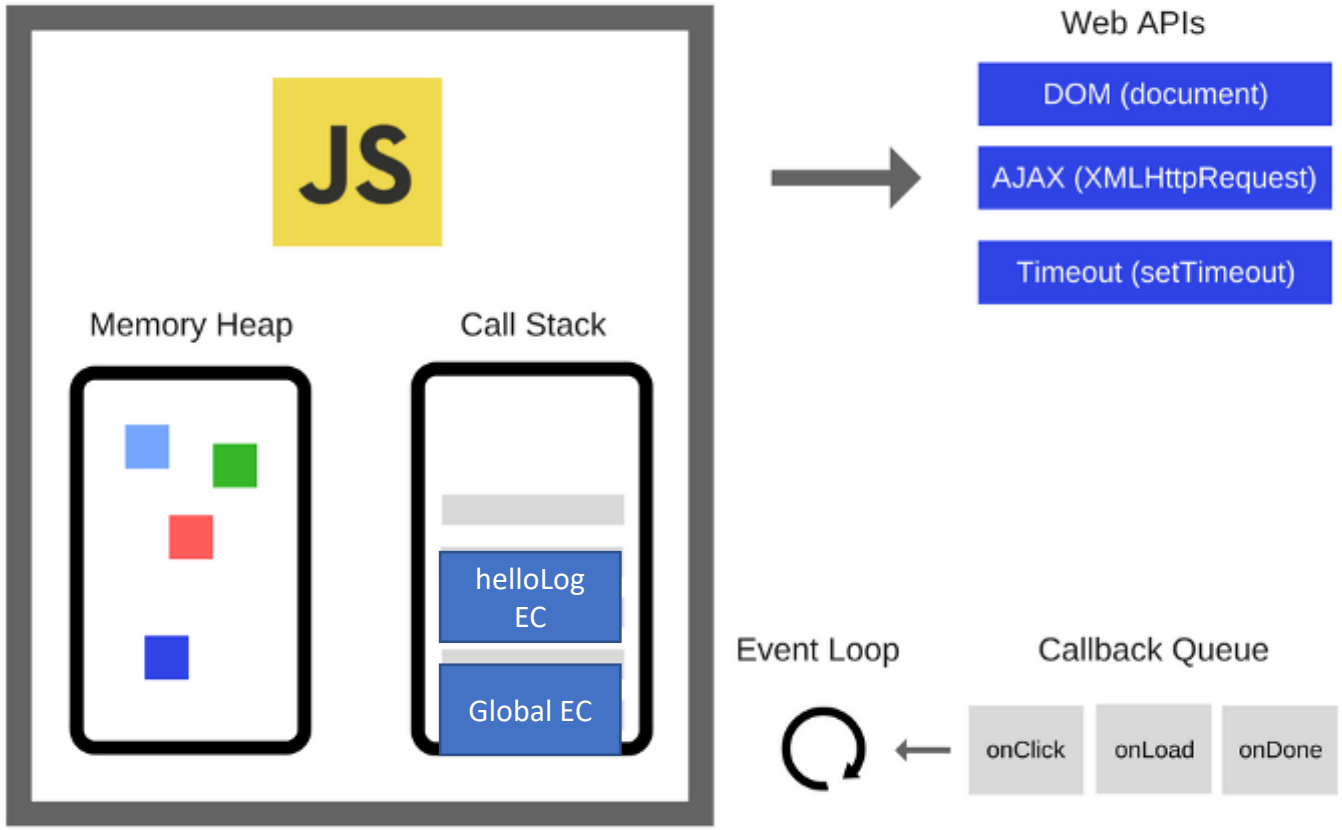
- JavaScript is a single-threaded, multi-paradigm, “prototype” object oriented language
  - Multi-paradigm: JavaScript can be run as a server-side application or a purely browser-based application

  What is the most commonly used JavaScript engine used to run JavaScript?
  - V8! V8 is an engine developed by Google, which Chrome uses as well as Node.js
  - So, typically people think of JavaScript as interpreted. This means it is not a compiled language, but instead something known as an interpreter will read the code line by line, and then figure out what to do
  - So, this is not strictly true in modern JavaScript, especially for a cutting edge engine like V8.
    - Instead, V8 does what is known as JIT (just-in-time) compilation, whereby it actually does do compilation on the fly and execute these bits of compiled code as we go along. This helps make JavaScript more performant

Why is JavaScript so Weird????

- One guy basically made JavaScript in 10 days
- He really didn’t think it would become the most used language for the web
- There were so many bugs and quirks with it, that just stuck over time and never got fixed
- In particular, one example would be the hoisting of “var” variables
- Which is just an artifact of the usefulness of hoisting functions

JavaScript Execution Environment



- Every function running has its own execution context
- Notice how in JS, we can have code that exists outside of functions
  - That code will be part of the Global EC

Every EC contains

- The variable environment
  - Let, const, and var
  - Functions
  - The “arguments” object
- The scope chain
  - For example, a nested function (a function inside another function), can access variables in the parent function as well as the global scope
- this keyword

Scoping

- Scopes in Java are “lexical”, meaning that the variable scopes are controlled by the placement of functions and blocks
  - Global scope
  - Function scope
  - Block scope: (let and const in ES6)
- The Scope chain keeps track of the chain of access that inner scopes have of the outer scopes
  - Everything that is nested deeper has access to everything in the outer

This keyword

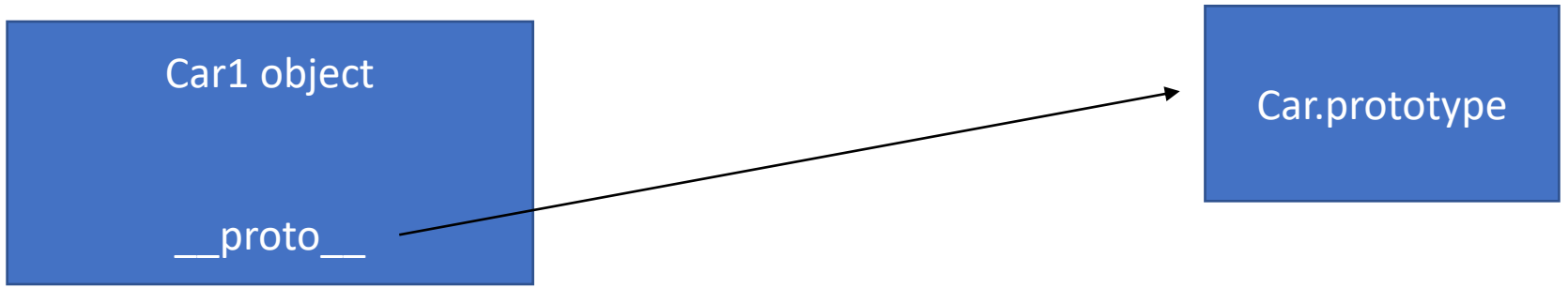
- This is a special variable that is created for each EC (aka every function)
- It points to what is calling the function
  - So, if we call the function without having an object calling the function, this will be undefined (in strict mode)
  - ‘this’ will refer to the window object in non-strict mode

Ways to declare functions

1. function() {}: a normal function declaration
2. let x = function() {}: anonymous function being assigned to a variable
3. () => {}: arrow functions
   - Arrow functions treat ‘this’ differently
   - It uses the ‘this’ from its parent EC, instead of having its own

Prototypal inheritance

- In JavaScript we have this idea of prototypal inheritance
  - This is different than the “class as a blueprint” and objects as an instance model of OOP
  - So, JavaScript does not follow the classical OOP like Java does
- Every object has a __proto__ property, which is linking to some other object from which methods can be used
  - In the case of function constructors, these functions have a prototype property that is an object w/ a collection of methods



HTML/CSS/JavaScript

- HTML is the noun
- CSS is the adjective
- JavaScript is the verb