JavaScript History

Asst.Prof. Dr. Umaporn Supasitthimethee
ผศ.ดร.อุมาพร สุภสิทธิเมธี

JavaScript History

- **1995** - JavaScript is a programming language that was created by **Brendan Eich** who was working for *Netscape.*
- **1997** - JavaScript 1.1 proposal was submitted to the European Computer Manufacturers Association (ECMA).

1 ชื่อแรก Mocha ชื่อก่อนเป็น JavaScript คือ ไลฟ์สคริปต์

มาตรฐานภาษา ที่ java scrip ไป compile

ECMAScript

- The formal specification of the JavaScript language specified in the document ***ECMA-262***
- ***ES1, ES2, ES3,…ESX*** are a different version of the ***ECMAScript*** specification

https://en.wikipedia.org/wiki/ECMAScript

* Started from ES6, version of the ECMAScript start naming the versions based on the year of published specification, for example, ES2015 (ES6),ES2016 (ES7), …

2 version ที่ทำให้ javascrip เป็นที่ยอมรับ และการมาของ node.js

JavaScript

2 ES5 ES6

ES5 (2009) is fully supported by most modern browser in early 2016 Fully support

- Higher-order iteration functions (map, reduce, filter, foreach);
- JSON support;
- Better reflection and object properties;

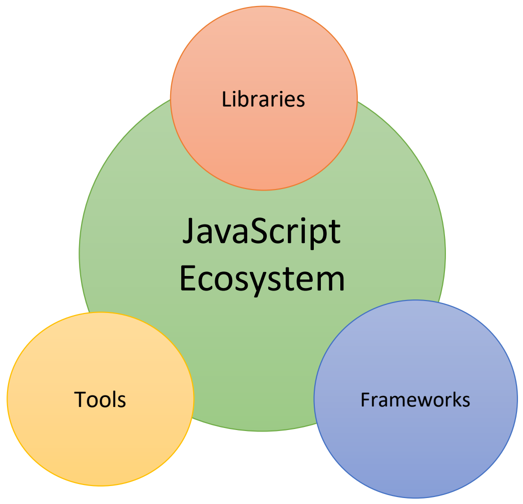
ES6 (ES2015) provide a greatly improved developer experience

- let, const
- Classes
- Modules
- Iterators
- Generators
- Promises
- Arrow functions

From 2016 to 2019, a new edition of the ECMAScript standard was published each year, but the scope of changes was much smaller than the 5th or 6th editions

Current Version: 14th Edition – ECMAScript 2023 (ES2023)

https://en.wikipedia.org/wiki/ECMAScript_version_history#Versions



JavaScript EcoSystem

Asst.Prof. Dr. Umaporn Supasitthimethee
ผศ.ดร.อุมาพร สุภสิทธิเมธี

The different aspects of JavaScript

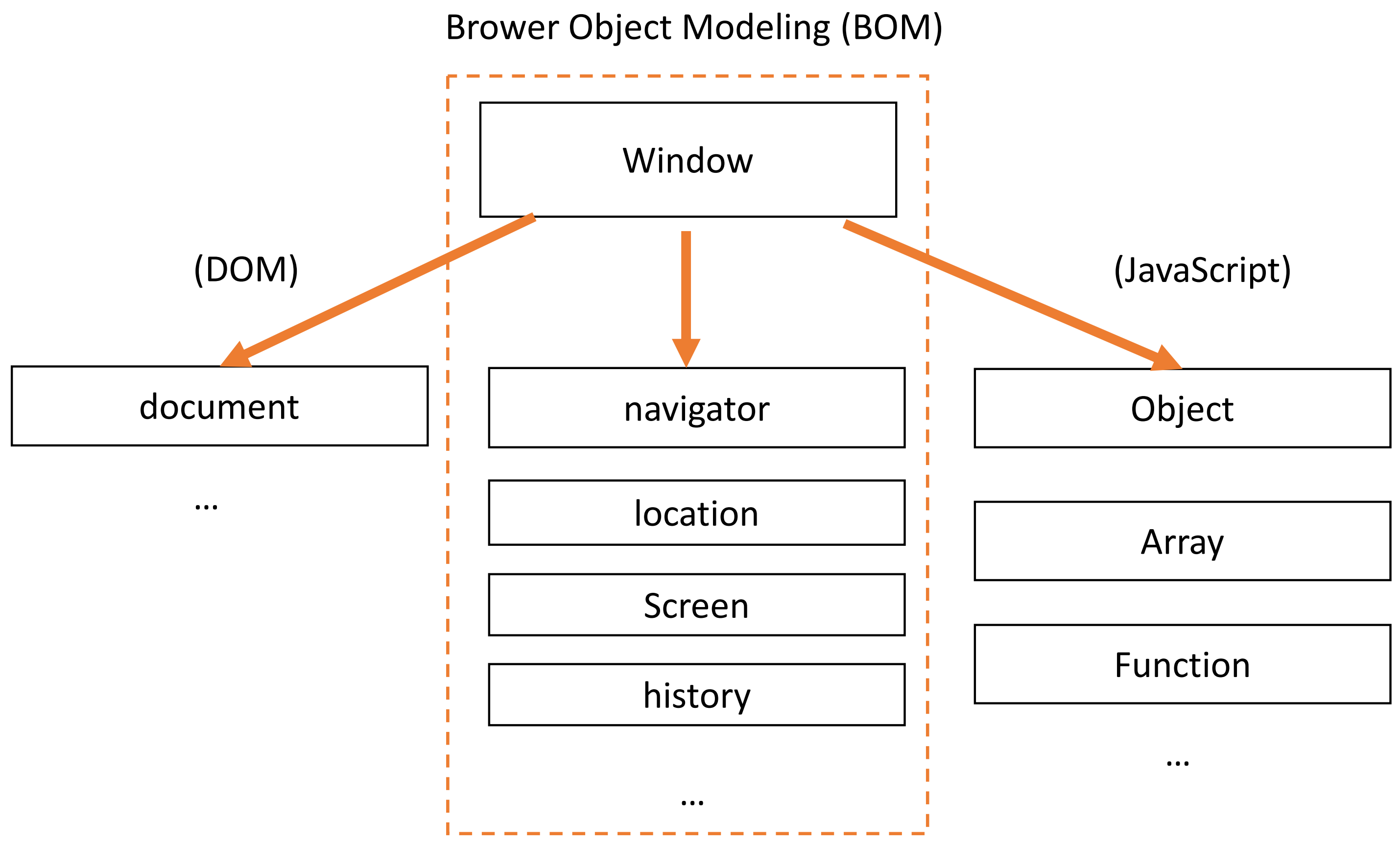
- Front-End: React, Angular, Vue.js, svelte, jQuery, NEXT.JS
- Back-End: node.js Deno, Bun
- Web Framework: Express, Nest.js
- Mobile: React Native, Apache Cordova Ionic
- Desktop: Electron
- Database: MongoDB

Introduction to JavaScript

- JavaScript is the programming language of the web.
- The overwhelming majority of websites use JavaScript, and all modern web browsers—on desktops, tablets, and phones
- Over the last decade, Node.js has enabled JavaScript programming outside of web browsers, and the dramatic success of Node means that JavaScript is now also the most-used programming language among software developers. ความสำเร็จอย่างมาก
- JavaScript is completely different from the Java programming language.

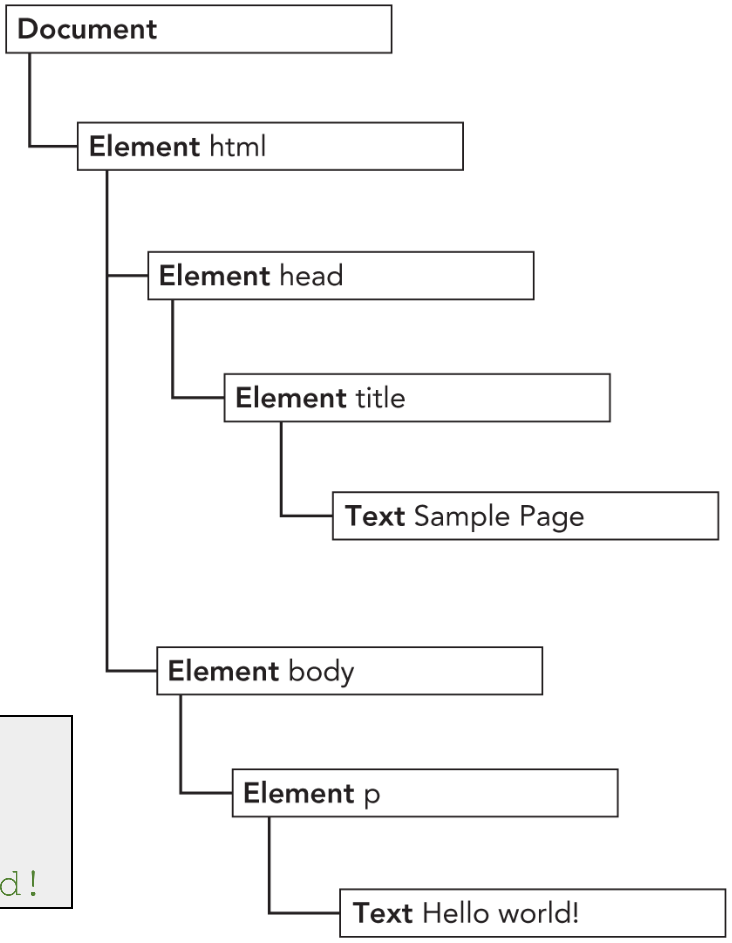
JavaScript Java

The Window interface represents a window containing a DOM document.
In a tabbed browser, each tab is represented by its own Window object.



DOM: The Document Object Model

```
<html>
   <head>
      <title>Sample Page</title>
   </head>
   <body>
      <p>Hello World!</p>
   </body>
</html>
```



```
const paragraphs =
document.getElementsByTagName("p");
alert(paragraphs[0].nodeName); //p
alert(paragraphs[0].textContent); //Hello World!
```

node.js JavaScript ?

ทำไมการมาของ node.js ถึงทำให้ javascript สำคัญขึ้น?

เพราะ สมัยก่อนไม่มี node การ run javascript ต้อง run ที่ browser เท่านั้น

แต่จุดเปลี่ยนจริงๆที่ไม่ต้อง run ที่ browser ก็ได้ คือ Chromium project กับ Chrome V8 (เป็น project ของ google)

Chromium

open source browser project

Web Browser

Chromium-based browser: Google Chrome Microsoft Edge Opera

Safari is a graphical web browser developed by Apple, based on the WebKit engine.

Mozilla Firefox, or simply **Firefox**, is a free and open-source web browser developed by the Mozilla Foundation and its subsidiary, the Mozilla Corporation. Firefox uses the Gecko layout engine to render web pages.

Chrome V8 คือ javascript engine ที่อยู่ใน Browser ข้างบน

Chrome V8

JavaScript Engine

open source JavaScript engine project

Chrome V8: Google Chrome Microsoft Edge Opera

JavaScriptCore: A JavaScript interpreter and JIT originally derived from KJS. It is used in the WebKit project and applications such as **Safari**.

SpiderMonkey: A JavaScript engine in Mozilla Gecko applications, including **Firefox.**

node.js ใช้ Chrome V8

JavaScript Development Environment

In Web Browser

- Google Chrome
- Microsoft Edge
- Safari
- Firefox
- Opera

Outside Web Browser (based on Chrome V8 JavaScript Engine)

Node.js: a JavaScript runtime built on Chrome's V8 JavaScript engine.

Deno: a simple, modern and secure runtime for JavaScript and TypeScript that uses Chrome's V8 and is built in Rust.

Bun: Bun is a fast all-in-one JavaScript runtime Bundle, transpile, install and run JavaScript & TypeScript projects — all in Bun.

Demo JavaScript

In and Outside Web Browser

MyFirstScript.js

```
console.log("I am JavaScript.");
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script src="MyFirstScript.js"></script>
</head>
<body>
    <h1>Hello, This is my HTML page with JavaScript.</h1>
</body>
</html>
```

Vanilla JavaScript

“Vanilla JavaScript” is just plain or pure JavaScript without any additional libraries or framework

เขียนเองทั้งหมด ไม่ได้ใช้พวก framework หรือ library

JS Introduction to JavaScript

Asst.Prof. Dr. Umaporn Supasitthimethee
ผศ.ดร.อุมาพร สุภสิทธิเมธี

JavaScript | MDN (mozilla.org)

JavaScript: The Definitive Guide, Seventh Edition, by David Flanagan

JavaScript Language Features

บรรทัดไหนผิดจะแจ้งทันที ไม่เหมือน complie

ทำ 1 งานได้ 1 เวลา ทำหลายงานไม่ได้ แต่มีเทคนิคพิเศษที่เสมือนว่าเป็น multi task ชื่อ event loop

- Interpreted Language
- Single Threaded, do one operation at one time
- Dynamically and weakly typed language
- Support Object Oriented Programming (Prototyped-based)

Dynamically และ weakly type คือเปลี่ยนแปลงได้ตลอดเวลา ให้ผลกับค่าล่าสุดเสมอ

inte

Asynchronous vs. Synchronous Programming

- **Synchronous** tasks are performed one at a time and only when one is completed, the following is unblocked. In other words, you need to wait for a task to finish to move to the next one.
- **Asynchronous** tasks can start, execute, and complete independently of each other. Instead of waiting for a task to finish before moving on, the program can continue executing other tasks while the asynchronous task is being processed. Once the asynchronous task is completed, a callback function or a promise is used to handle the result. ถ้ามีงานไหนที่ต้องใช้เวลา มันจะเอางานอื่นขึ้นมาทำด้วย

Res

Asynchronous Callback Functions

In JavaScript, a callback function is a function that is passed into another function as an argument.

This function can then be invoked during the execution of that higher order function.

```
console.log('Hello’);

setTimeout(function () {
  console.log('JS');
}, 5000);

console.log('Bye bye');
```

```
//Console

Hello
Bye bye

//until 5 seconds
JS
```

`setTimeout()` executes a particular block of code once after a specified time has elapsed.

Common mechanisms used for handling asynchronous programming in JavaScript include callbacks, promises, and async/await.

Higher-Order Functions

A “higher-order function” is a function that accepts functions as parameters and/or returns a function.

- JavaScript Functions are **first-class citizens**
  - be assigned to variables (and treated as a value)
  - be passed as an argument of another function
  - be returned function as a value from another function

```
//1. store functions in variables

function add(n1, n2) {
  return n1 + n2
}
let sum = add

let addResult1 = add(10, 20)
let addResult2 = sum(10, 20)

console.log(`add result1: ${addResult1}`)
console.log(`add result2: ${addResult2}`)
```

```
//2. Passing a function to another function
function operator(n1, n2, fn) {
  return fn(n1, n2)
}
function multiply(n1, n2) {
  return n1 * n2
}

let addResult3 = operator(5, 3, add)
let multiplyResult = operator(5, 3, multiply)

console.log(`add result3 : ${addResult3}`)
console.log(`multiply result: ${multiplyResult}`)
```

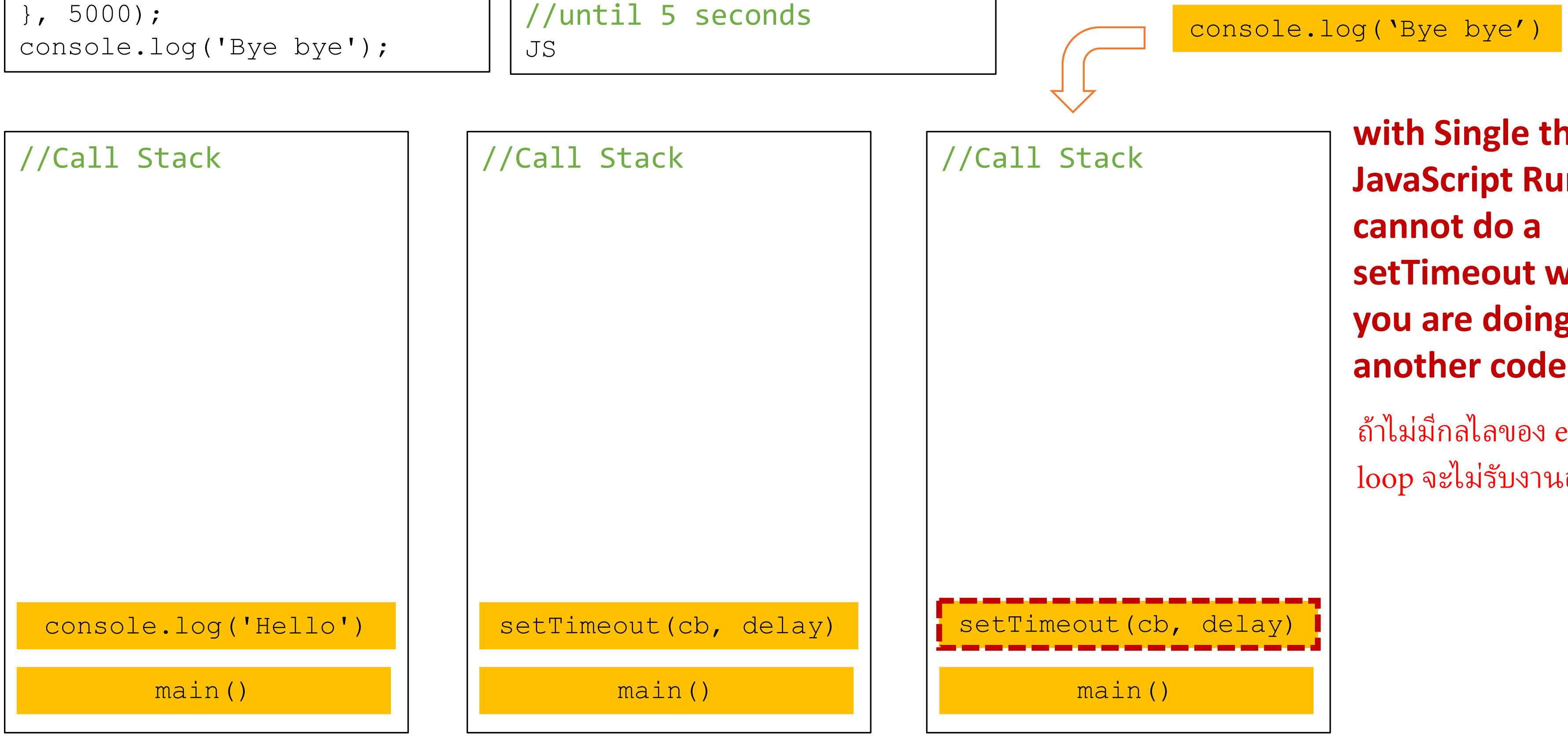
```
//3. return function as value of another function
function sayGoodBye(){
    return 'Good bye'
}
function doSomething(){
    return sayGoodBye
}
let doIt=doSomething()
console.log(doIt())
```

Stack เข้าก่อนออกที่หลัง

```
console.log('Hello’);
setTimeout(function cb() {
  console.log('JS');
}, 5000);
console.log('Bye bye');
```

```
//Console
Hello
Bye bye
//until 5 seconds
JS
```

Synchronous programming



with Single thread, JavaScript Runtime cannot do a setTimeout while you are doing another code

ถ้าไม่มีกลไลของ event loop จะไม่รับงานอื่นเลย

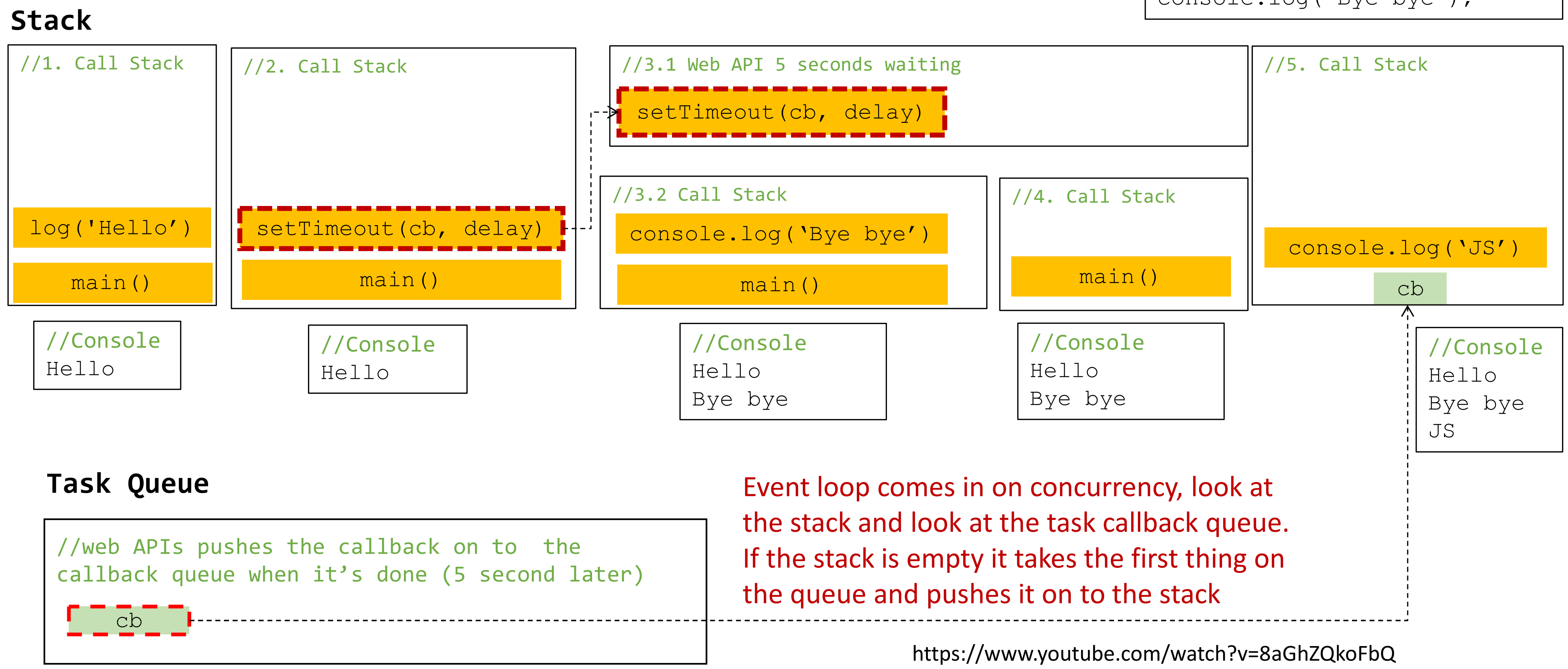
https://www.youtube.com/watch?v=8aGhZQkoFbQ

Stack เข้าก่อนออกที่หลัง
Queue มาก่อนได้ก่อน

JavaScript uses Event Loop and Callback Queue to work concurrently

//Example JS Program

```
console.log('Hello');
setTimeout(function cb() {
  console.log('JS');
}, 5000);
console.log('Bye bye');
```



Event loop comes in on concurrency, look at the stack and look at the task callback queue. If the stack is empty it takes the first thing on the queue and pushes it on to the stack

https://www.youtube.com/watch?v=8aGhZQkoFbQ

Event loop คือ ติดตามงานที่เป็น asyncornous | การทำงานหลายๆงานพร้อมกัน และเฝ้าติดตามงานที่กำลังรอคอยอยู่ เมื่อไหร่ที่งานเสร็จ จะเข้า Queue จากนั้นรอให้ Stack ว่างก่อน ถึงจะ push ขึ้น stack ได้