JS Introduction to JavaScript

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

JavaScript | MDN (mozilla.org)

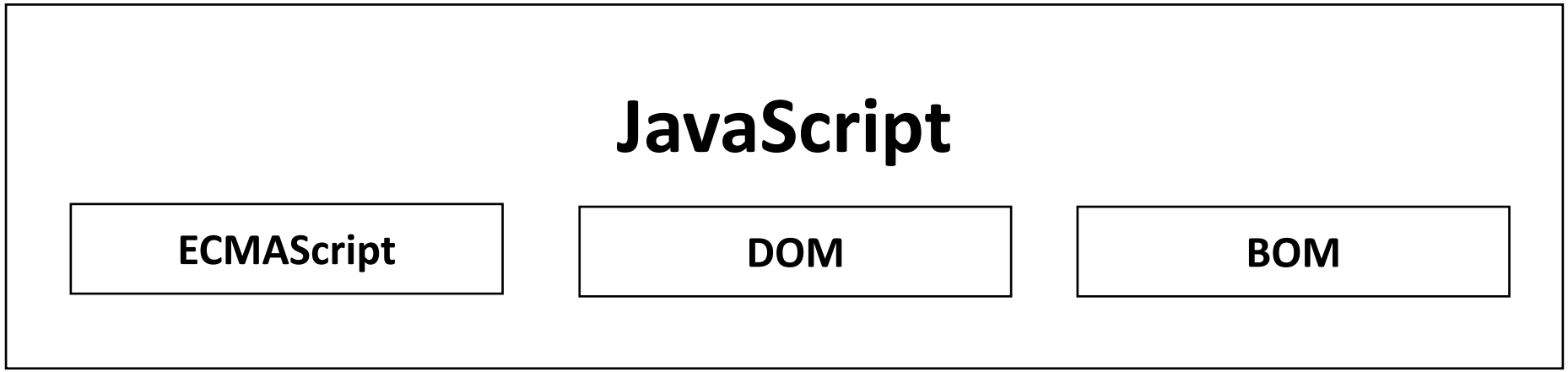
JavaScript: The Definitive Guide, Seventh Edition, by David Flanagan

Download Source Codes:
https://github.com/umaporn-sup/JS-SourceCodes-17082021.git

JavaScript Language Features

- Interpreted Language
- Single Threaded, do one operation at one time
- Dynamically and weakly typed language //02_TypesValuesVariables/script1.js
- Support Object Oriented Programming (Prototyped-based)

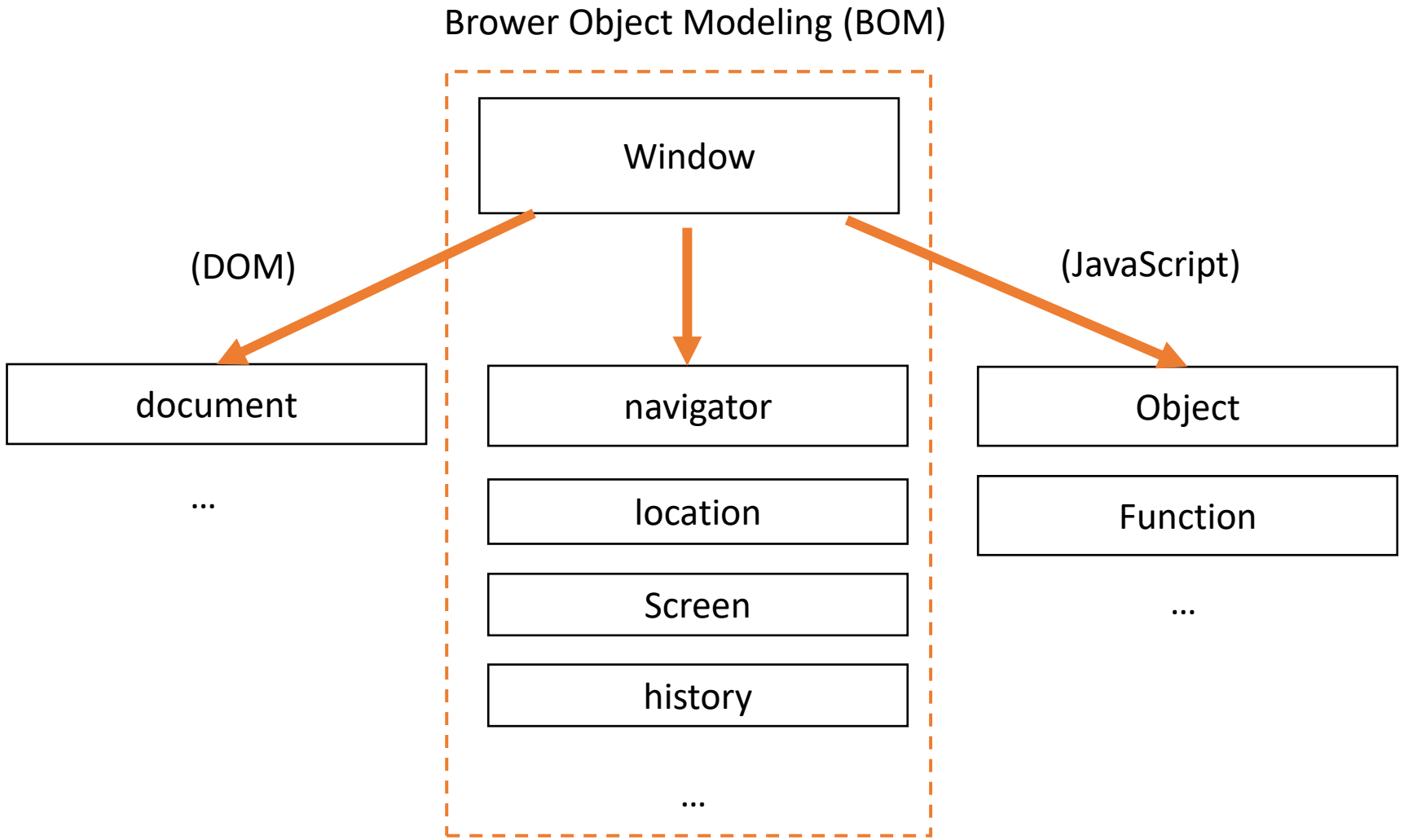
JavaScript



DOM: The Document Object Model
Map out an entire page as a hierarchy of nodes

BOM: The Browser Object Model
Deals with the browser window and frames

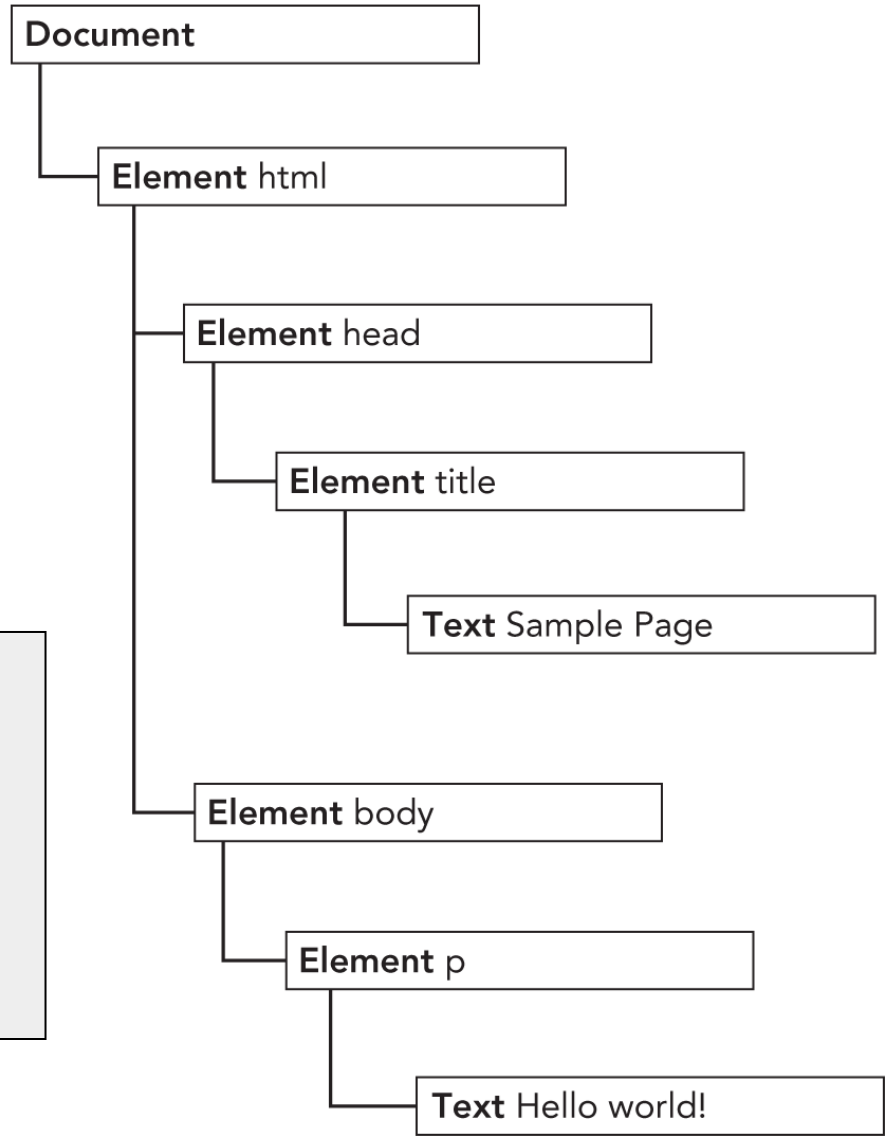
The `Window` interface represents a window containing a DOM document.
In a tabbed browser, each tab is represented by its own Window object.



DOM: The Document Object Model

//01_BasicJS/script3.js

```
<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].nodeValue);//null
```

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** software design expands upon the concept by building code that allows a program to ask that a task be performed alongside the original task (or tasks), **without stopping to wait for the task to complete**. When the secondary task is completed, the original task is notified using an agreed-upon mechanism so that it knows the work is done, and that the result, if any, is available.

https://developer.mozilla.org/en-US/docs/Glossary/Asynchronous

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.

Since, in JavaScript, functions are objects, functions can be passed as arguments.

// 01_BasicJS/script4.js

```
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.

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 as a value from another function

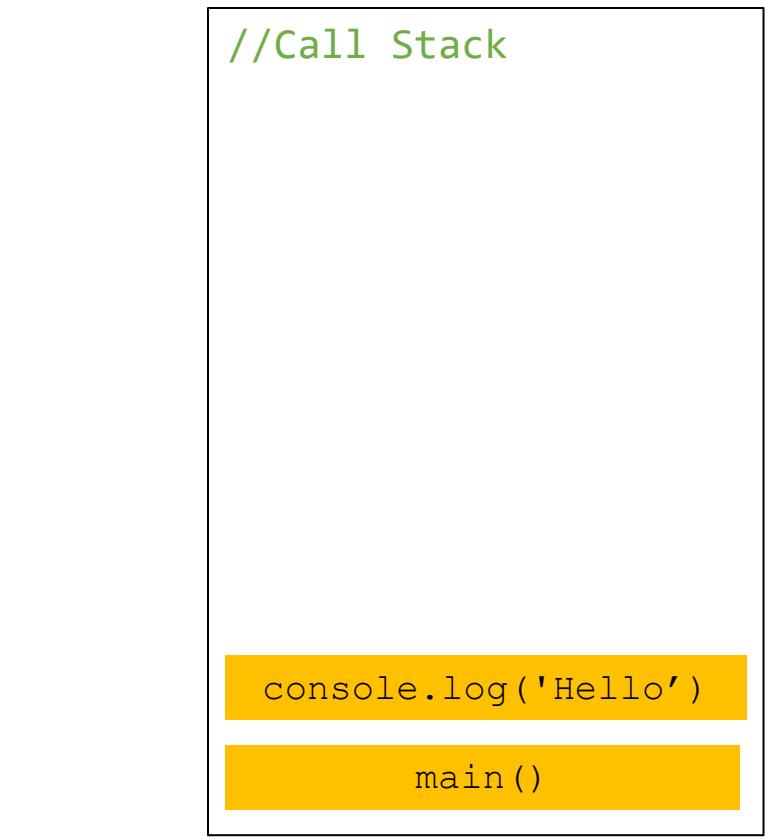
```
//1. store functions in variables

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}`)
```

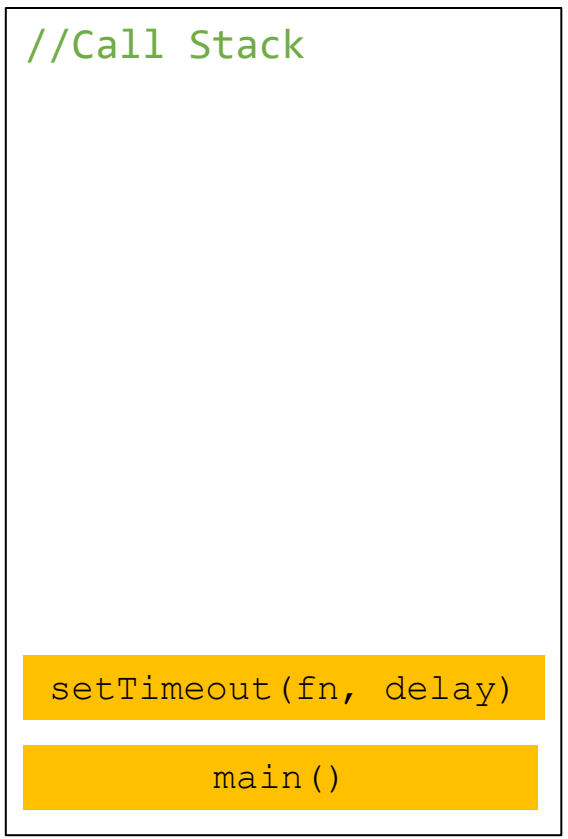
// 01_BasicJS/ script2.js

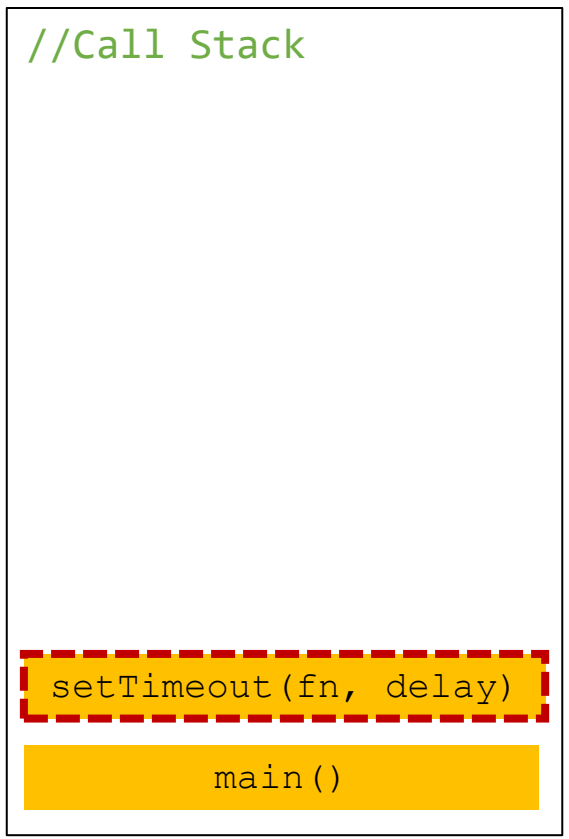
```
//2. returned as a value from another function

function operator(n1, n2, fn) {
  return fn(n1, n2)
}

//3. Passing a function to another function
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}`)
```

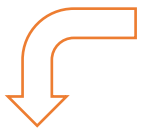
```
console.log('Hello');
setTimeout(function () {
  console.log('JS');
}, 5000);
console.log('Bye bye');
```

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









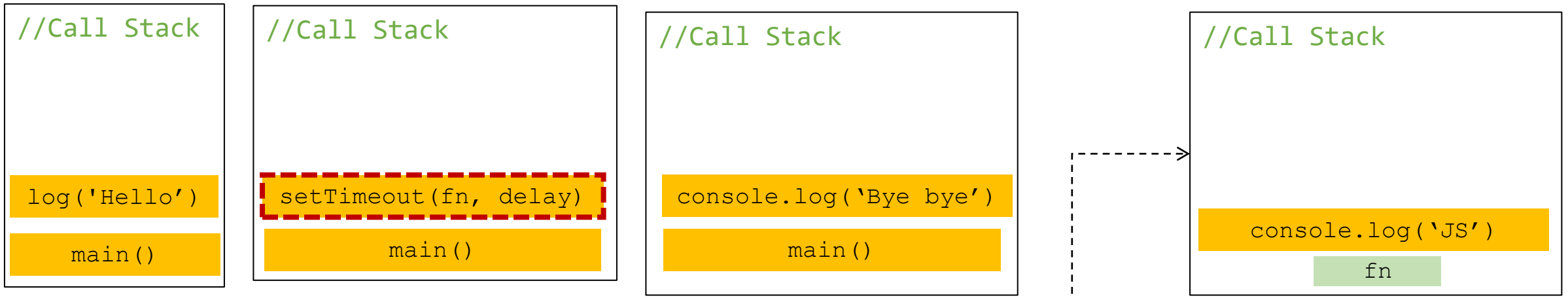
`console.log('Bye bye')`

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

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

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

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



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

callback queue

```
//web APIs pushes the callback on to  the
callback queue when it's done
  fn
```

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

Vanilla JavaScript

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