# JAVASCRIPT

By: Michel Samir Zaki

# JS in Web Development

JavaScript is **lightweight , cross platform and object oriented** programming language

javaScript is one of **three core technologies** of web development (**HTML – CSS - JS**)

# JS in Web Development

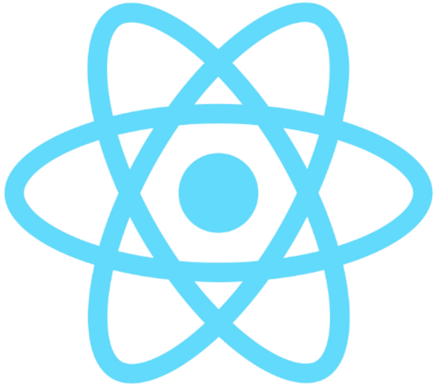
**For web development:**

- javaScript is traditionally only been used in browser environment (client-side language)
- lately: thanks to Node.JS we can use javascript on server-side

# JS in Web Development Cont.

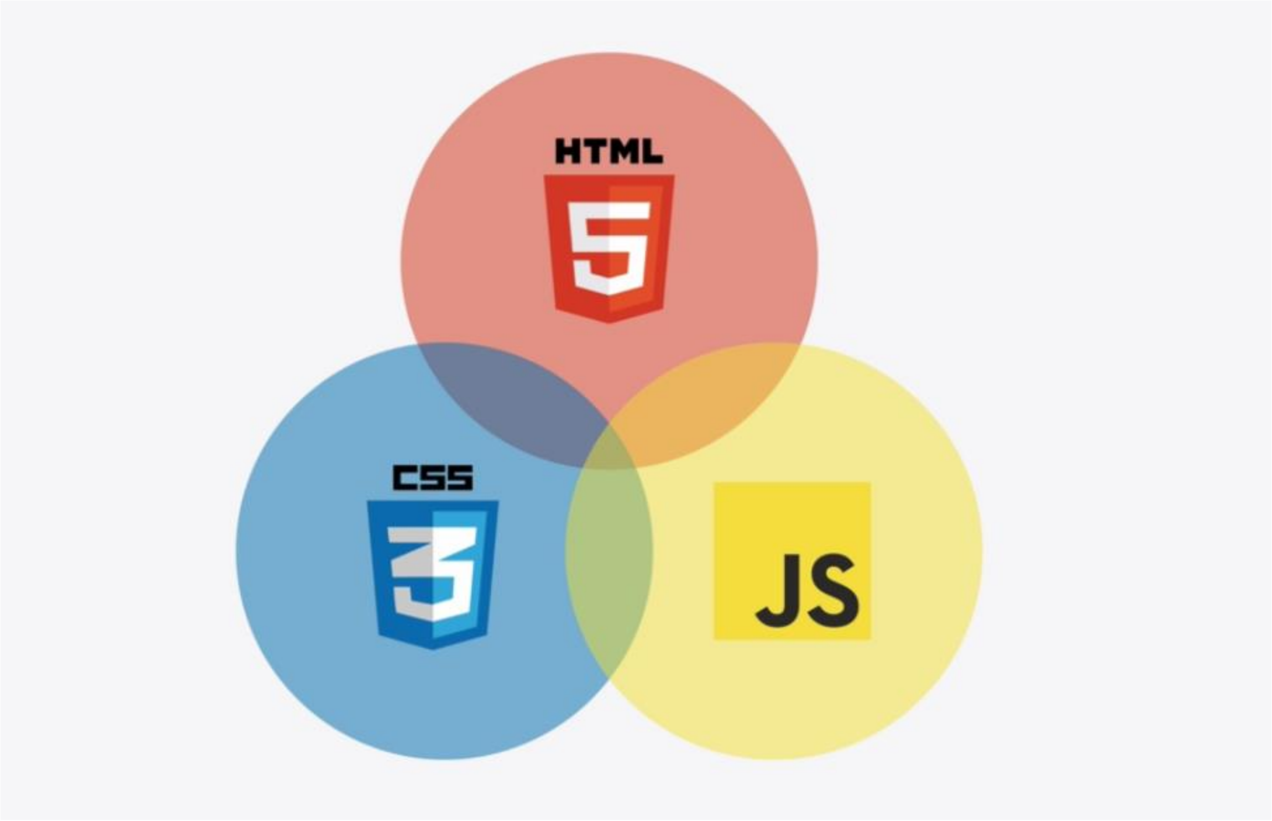
JavaScript allows us to develop complex modern web applications that we can interact with

According to **Angular** (javaScript framework), you need to master javaScript in order to use Angular

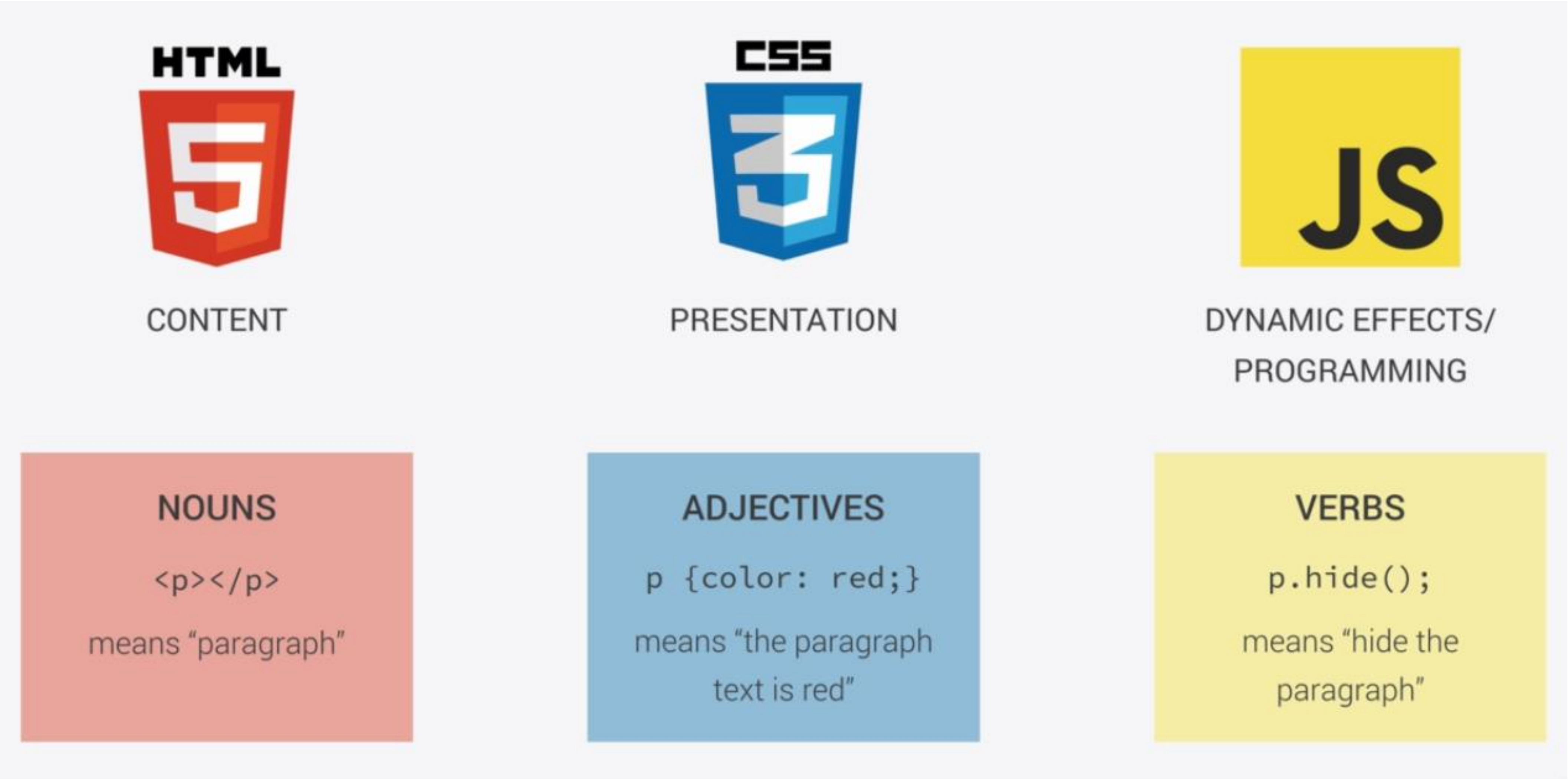


# JS in Web Development Cont.

all work together to create beautiful interactive and dynamic web applications



# JS in Web Development Cont.



HTML

CONTENT

**NOUNS**

`<p></p>`

means "paragraph"

CSS

PRESENTATION

**ADJECTIVES**

`p {color: red;}`

means "the paragraph text is red"

JS

DYNAMIC EFFECTS/ PROGRAMMING

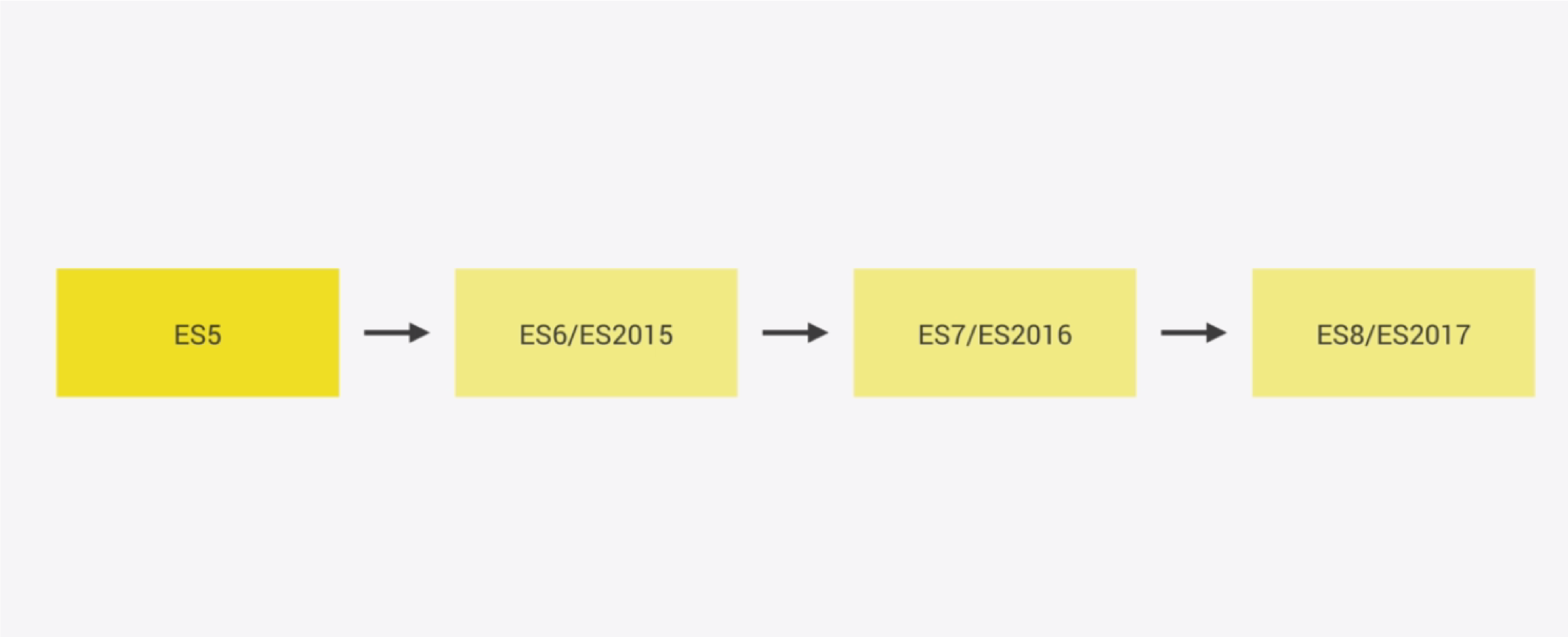
**VERBS**

`p.hide();`

means "hide the paragraph"

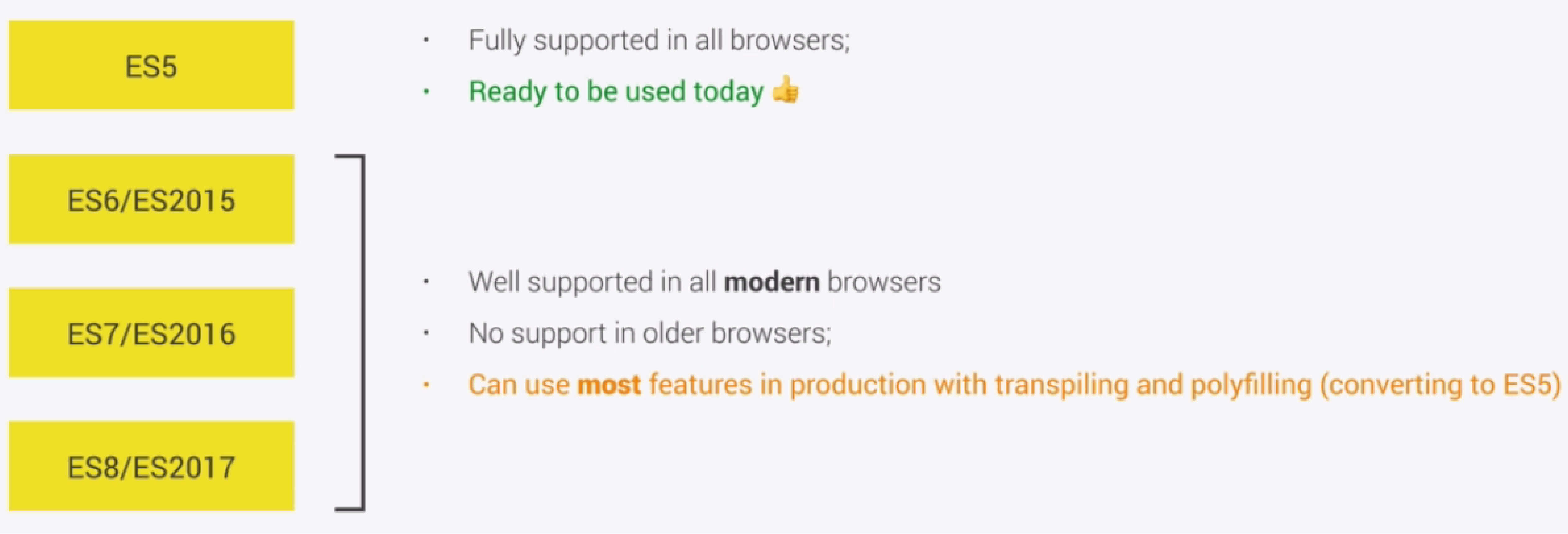
# JS Versions

- ECMAScript (Standard)



ES5 → ES6/ES2015 → ES7/ES2016 → ES8/ES2017

# JS Versions



ES5

- Fully supported in all browsers;
- Ready to be used today 👍

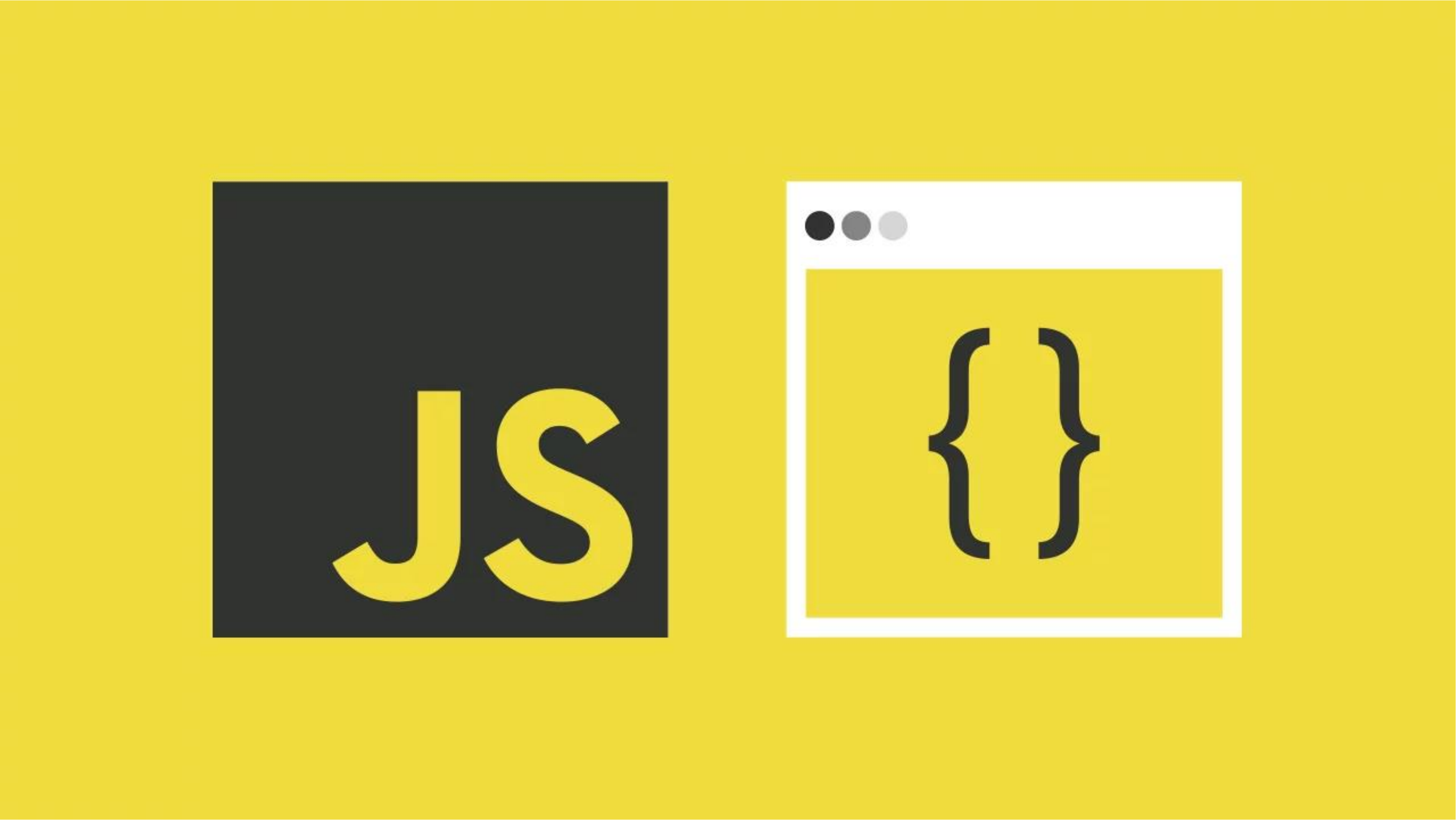
ES6/ES2015

ES7/ES2016

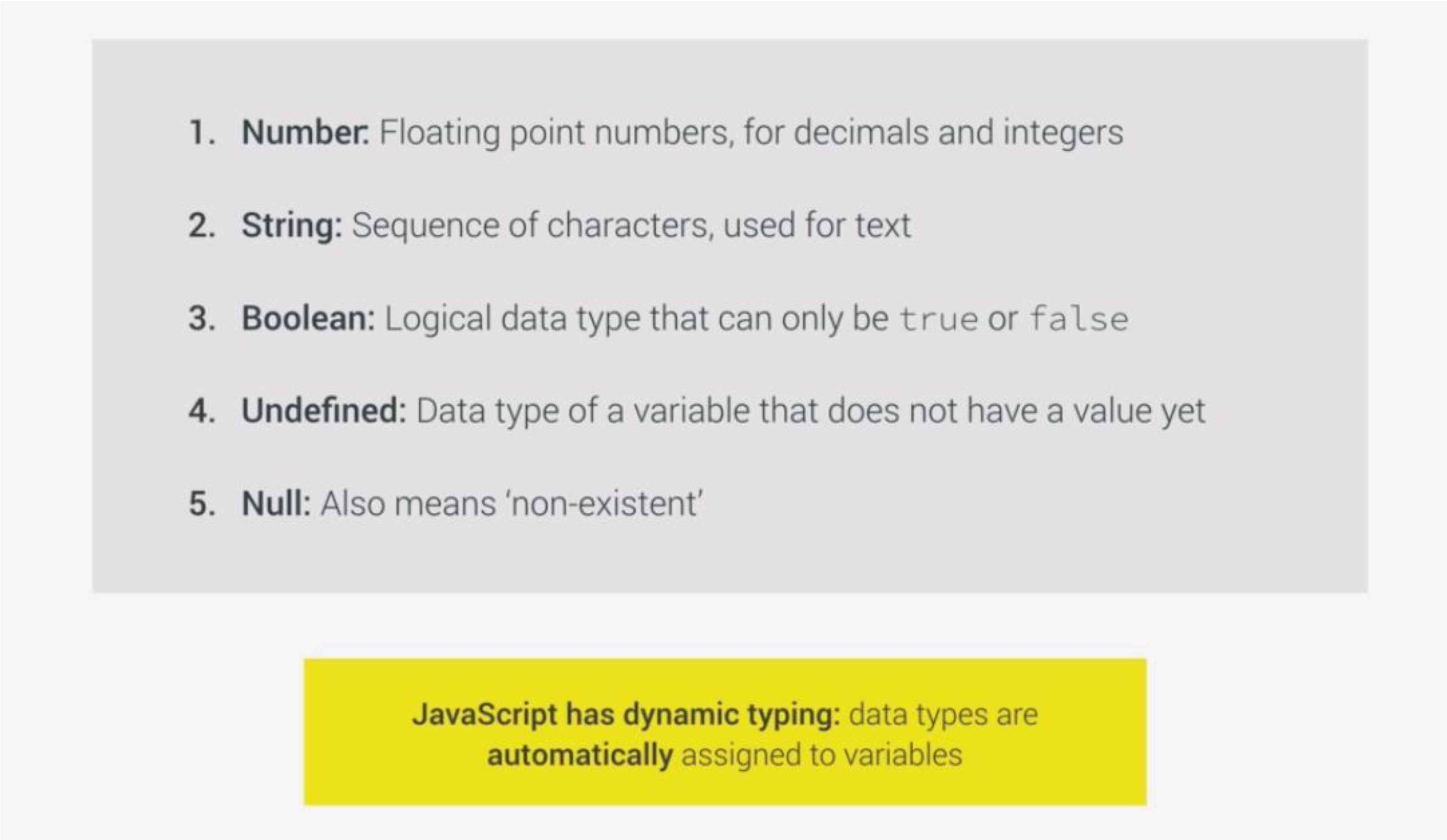
ES8/ES2017

- Well supported in all **modern** browsers
- No support in older browsers;
- Can use **most** features in production with transpiling and polyfilling (converting to ES5)

# Practice



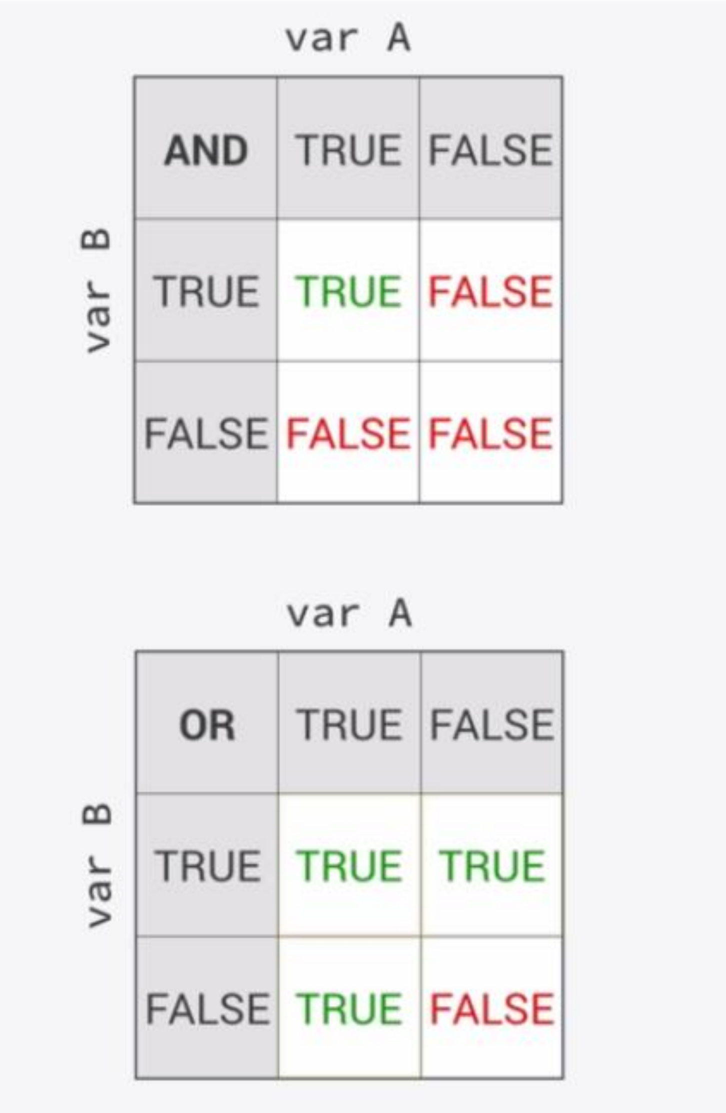
# Primitive data types



1. **Number:** Floating point numbers, for decimals and integers
2. **String:** Sequence of characters, used for text
3. **Boolean:** Logical data type that can only be `true` or `false`
4. **Undefined:** Data type of a variable that does not have a value yet
5. **Null:** Also means 'non-existent'

**JavaScript has dynamic typing:** data types are **automatically** assigned to variables

# logic operators



var A

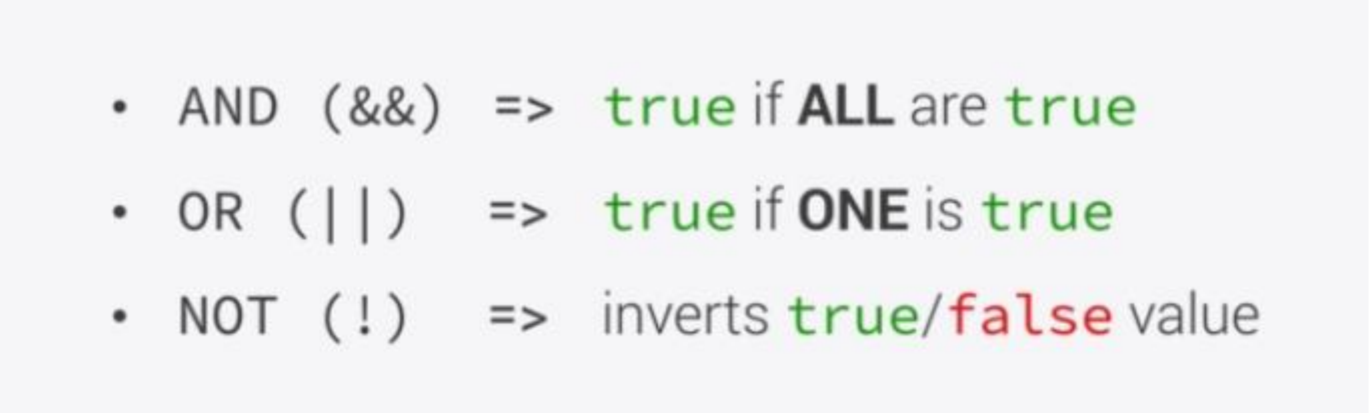
var B

| AND | TRUE | FALSE |
|---|---|---|
| TRUE | TRUE | FALSE |
| FALSE | FALSE | FALSE |

var A

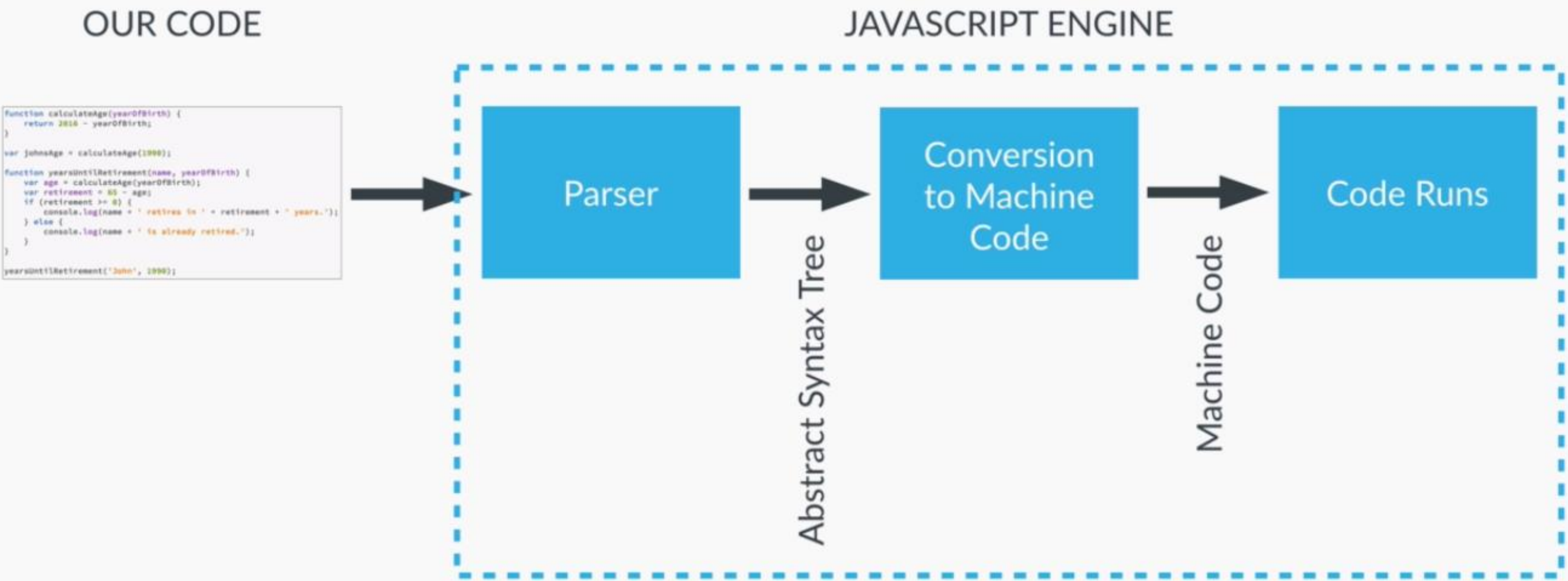
var B

| OR | TRUE | FALSE |
|---|---|---|
| TRUE | TRUE | TRUE |
| FALSE | TRUE | FALSE |



- AND (&&) => true if **ALL** are true
- OR (||) => true if **ONE** is true
- NOT (!) => inverts true/false value

# How Our Code Is Executed



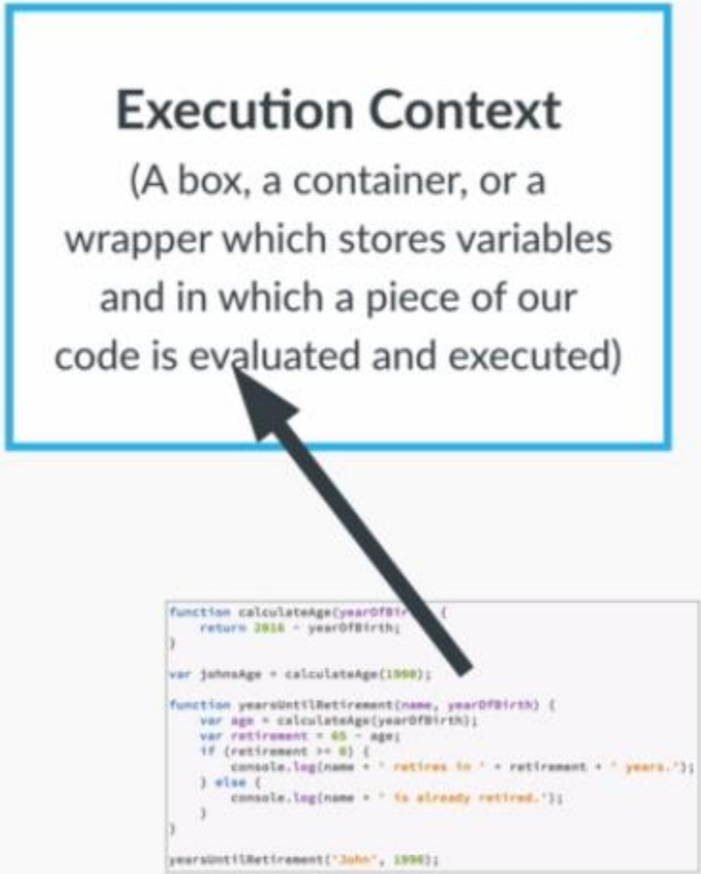
```
function calculateAge(yearOfBirth) {
    return 2016 - yearOfBirth;
}

var johnsAge = calculateAge(1990);

function yearsUntilRetirement(name, yearOfBirth) {
    var age = calculateAge(yearOfBirth);
    var retirement = 65 - age;
    if (retirement >= 0) {
        console.log(name + ' retires in ' + retirement + ' years.');
    } else {
        console.log(name + ' is already retired.');
    }
}

yearsUntilRetirement('John', 1990);
```

# How Our Code Is Executed



**Execution Context**

(A box, a container, or a wrapper which stores variables and in which a piece of our code is evaluated and executed)

```javascript
function calculateAge(yearOfBirth) {
    return 2016 - yearOfBirth;
}

var johnsAge = calculateAge(1990);

function yearsUntilRetirement(name, yearOfBirth) {
    var age = calculateAge(yearOfBirth);
    var retirement = 65 - age;
    if (retirement >= 0) {
        console.log(name + ' retires in ' + retirement + ' years.');
    } else {
        console.log(name + ' is already retired.');
    }
}

yearsUntilRetirement('John', 1990);
```

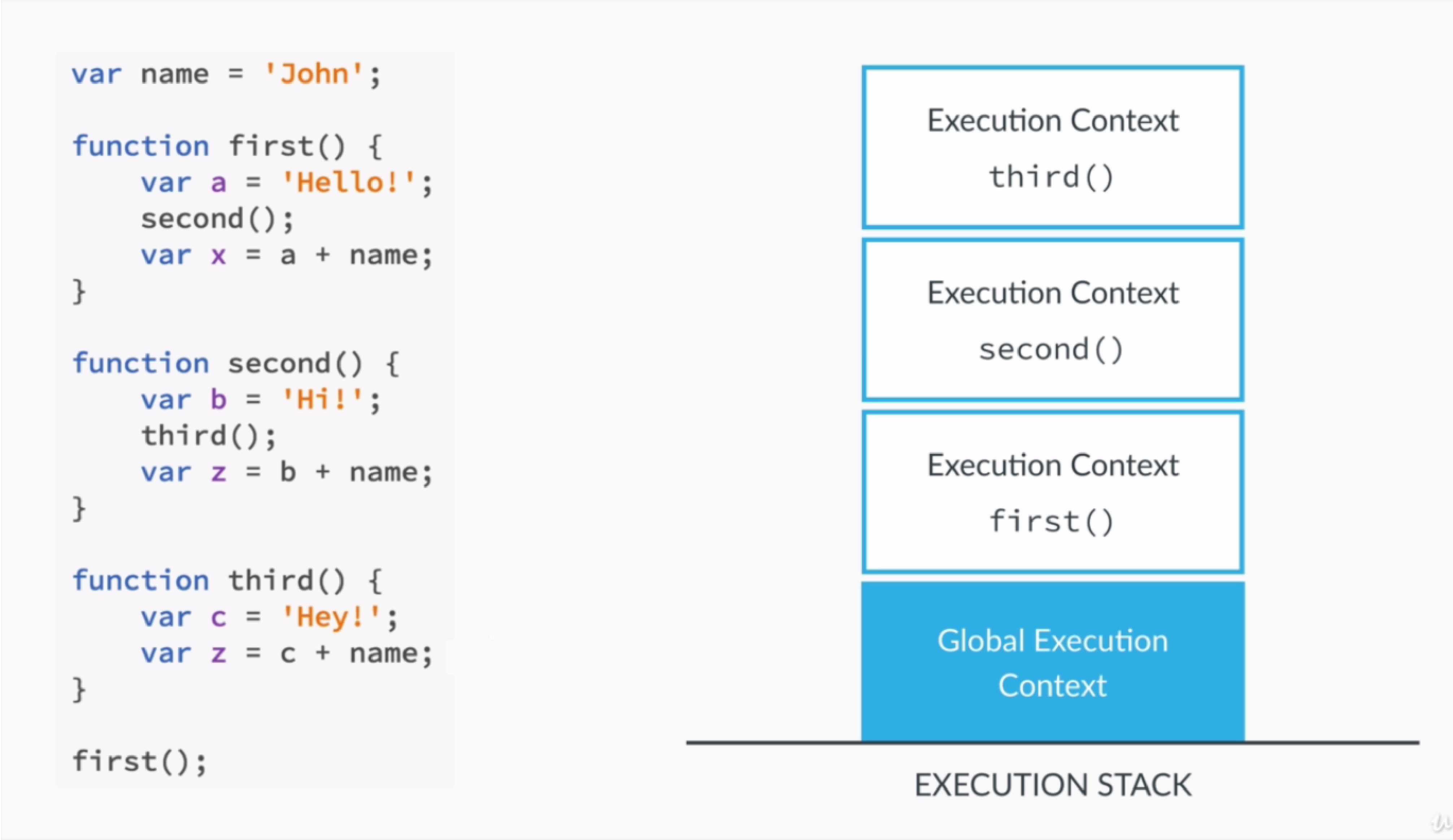
THE DEFAULT

**Global Execution Context**

- Code that is **not inside any function**
- Associated with the **global object**
- In the browser, that's the `window` object

```
lastName === window.lastName
// true
```

# How Our Code Is Executed



```
var name = 'John';

function first() {
    var a = 'Hello!';
    second();
    var x = a + name;
}

function second() {
    var b = 'Hi!';
    third();
    var z = b + name;
}

function third() {
    var c = 'Hey!';
    var z = c + name;
}

first();
```

Execution Context
third()

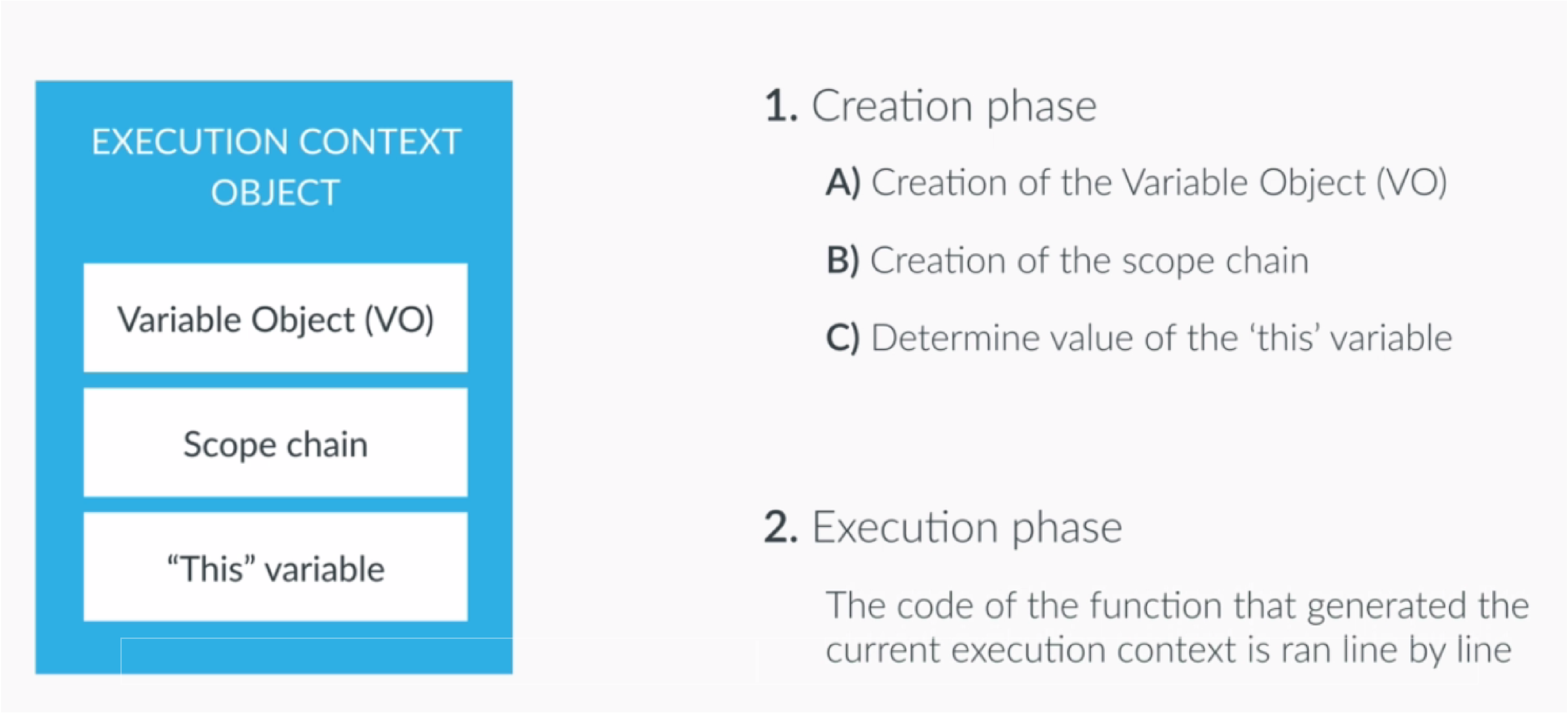
Execution Context
second()

Execution Context
first()

Global Execution Context

EXECUTION STACK

# How Our Code Is Executed



EXECUTION CONTEXT OBJECT

Variable Object (VO)

Scope chain

“This” variable

1. Creation phase

   A) Creation of the Variable Object (VO)

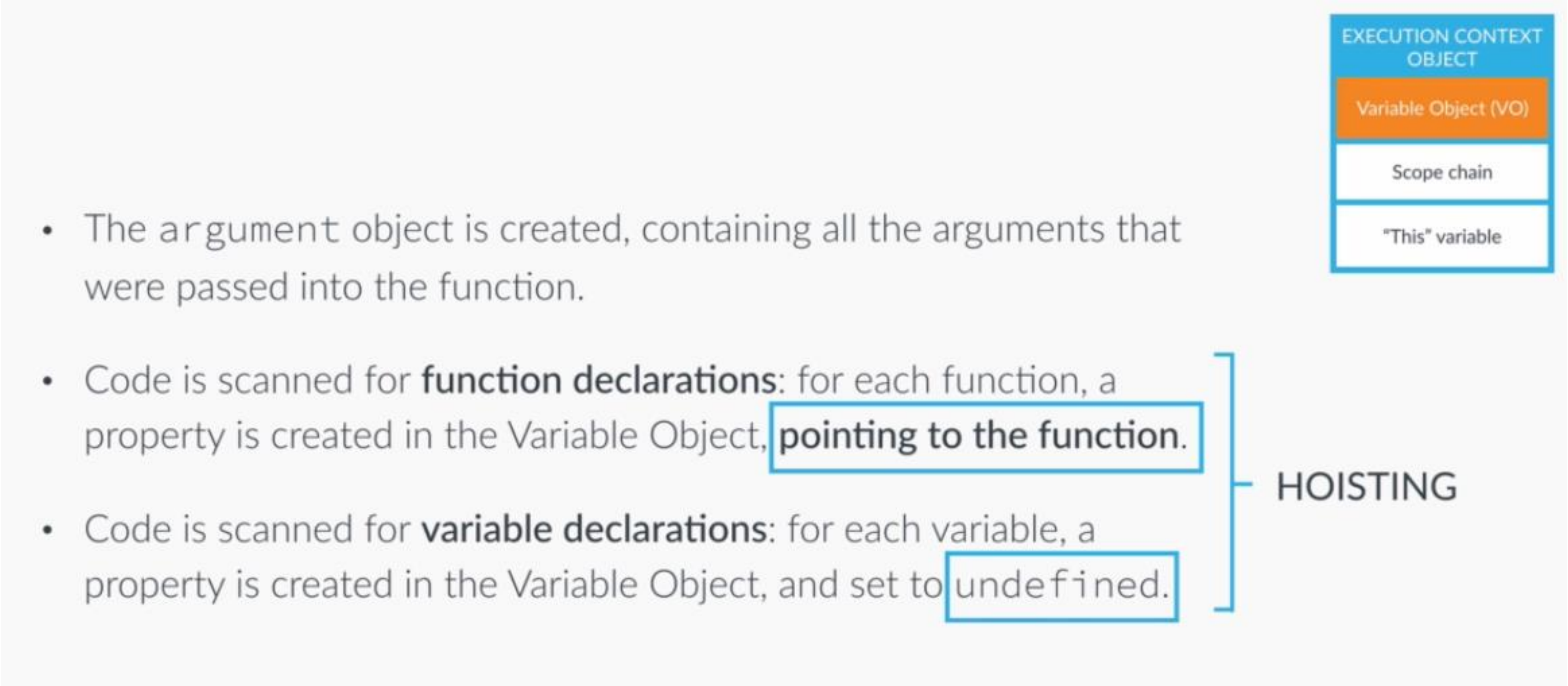
   B) Creation of the scope chain

   C) Determine value of the ‘this’ variable

2. Execution phase

   The code of the function that generated the current execution context is ran line by line

# How Our Code Is Executed



- The `argument` object is created, containing all the arguments that were passed into the function.
- Code is scanned for **function declarations**: for each function, a property is created in the Variable Object, **pointing to the function**.
- Code is scanned for **variable declarations**: for each variable, a property is created in the Variable Object, and set to `undefined`.

HOISTING

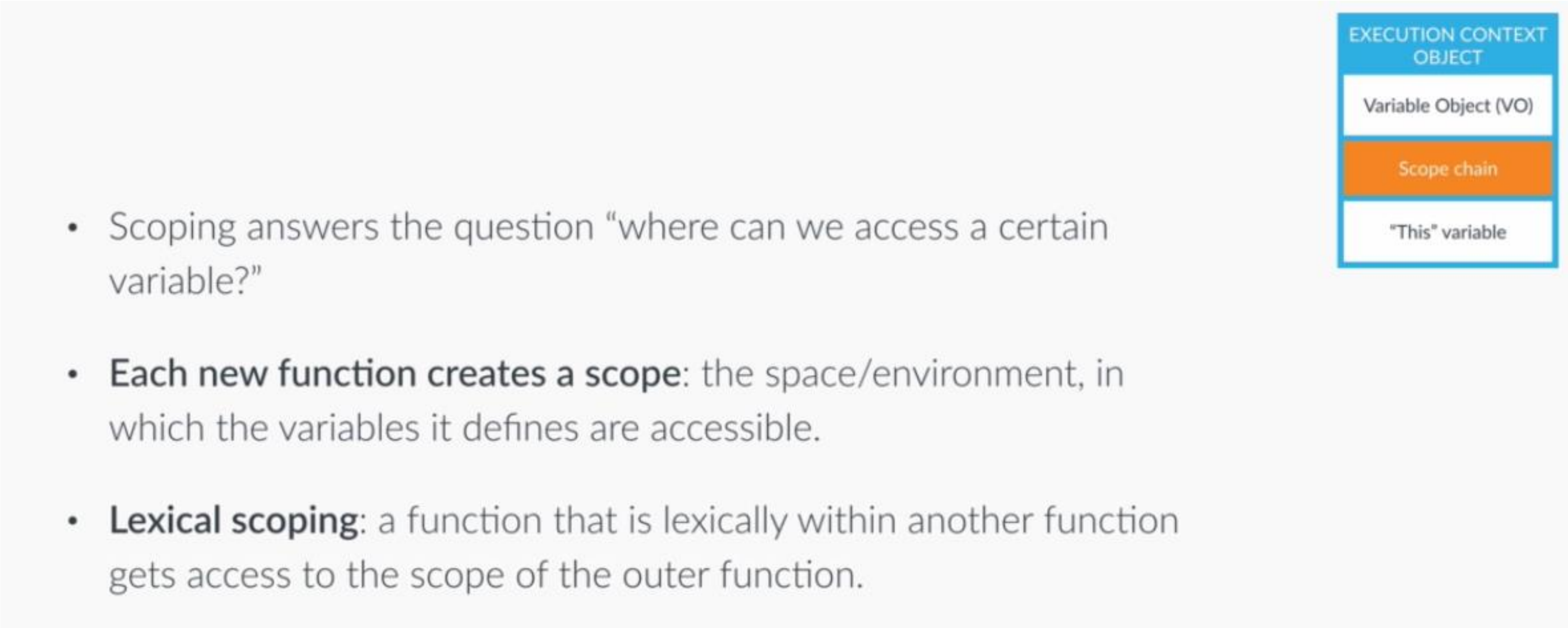
EXECUTION CONTEXT OBJECT

Variable Object (VO)

Scope chain

"This" variable

# How Our Code Is Executed



EXECUTION CONTEXT OBJECT

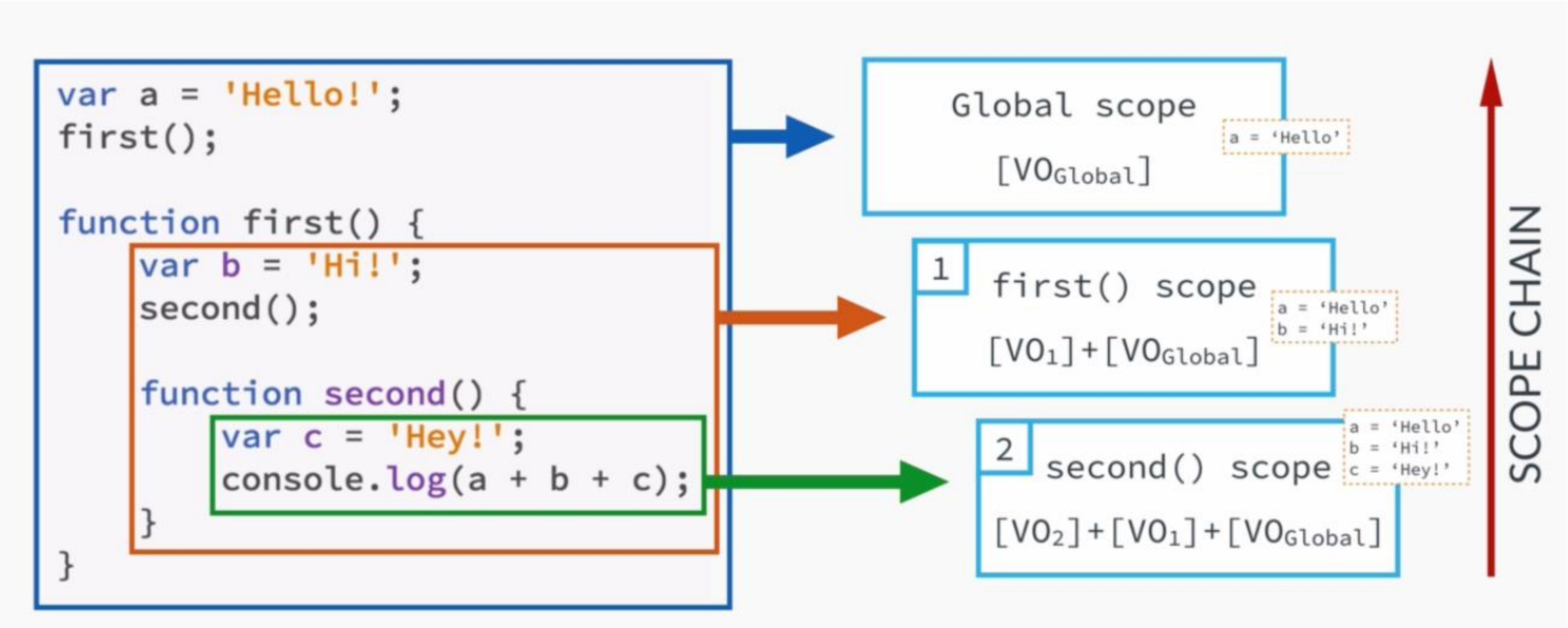
Variable Object (VO)

Scope chain

"This" variable

- Scoping answers the question "where can we access a certain variable?"
- **Each new function creates a scope**: the space/environment, in which the variables it defines are accessible.
- **Lexical scoping**: a function that is lexically within another function gets access to the scope of the outer function.

# How Our Code Is Executed

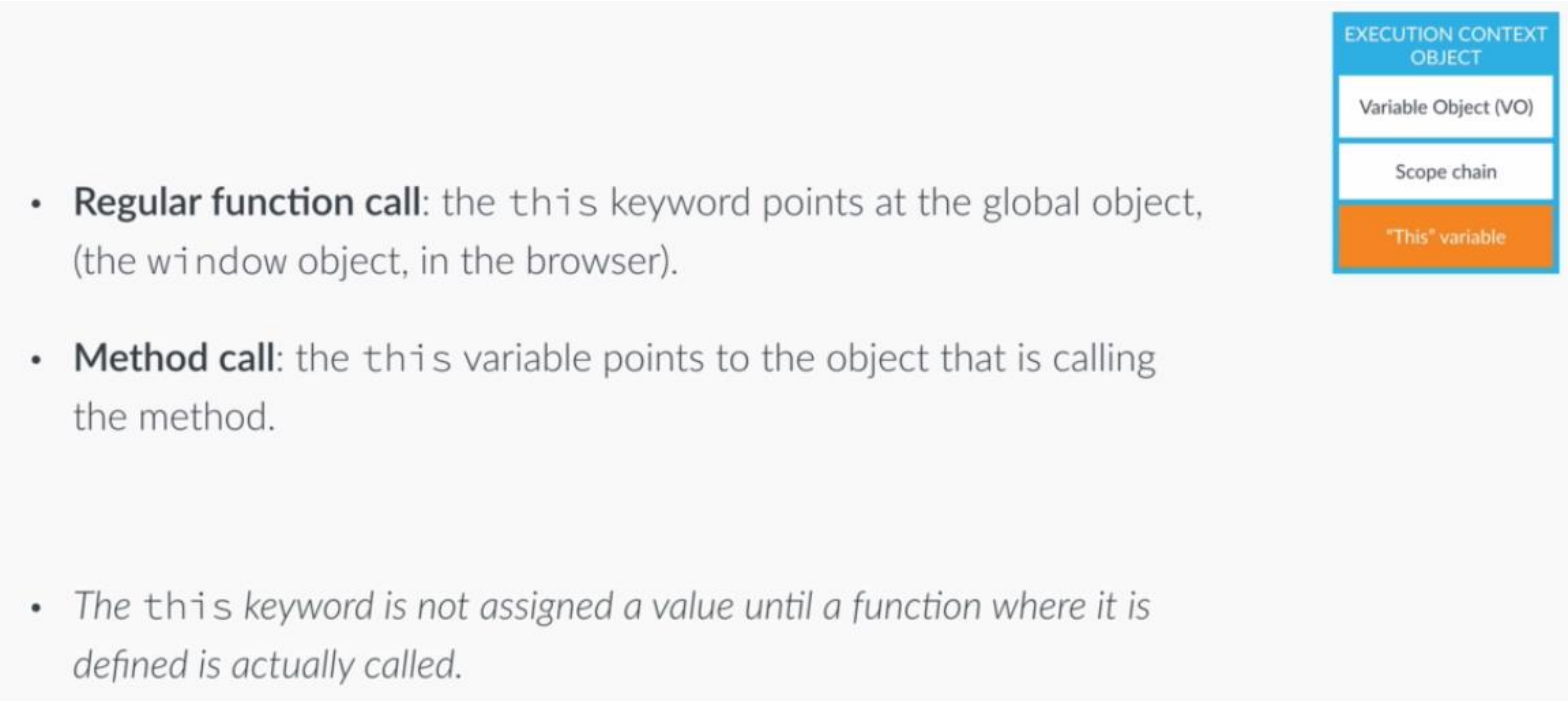


```
var a = 'Hello!';
first();

function first() {
    var b = 'Hi!';
    second();

    function second() {
        var c = 'Hey!';
        console.log(a + b + c);
    }
}
```

# How Our Code Is Executed



EXECUTION CONTEXT OBJECT

Variable Object (VO)

Scope chain

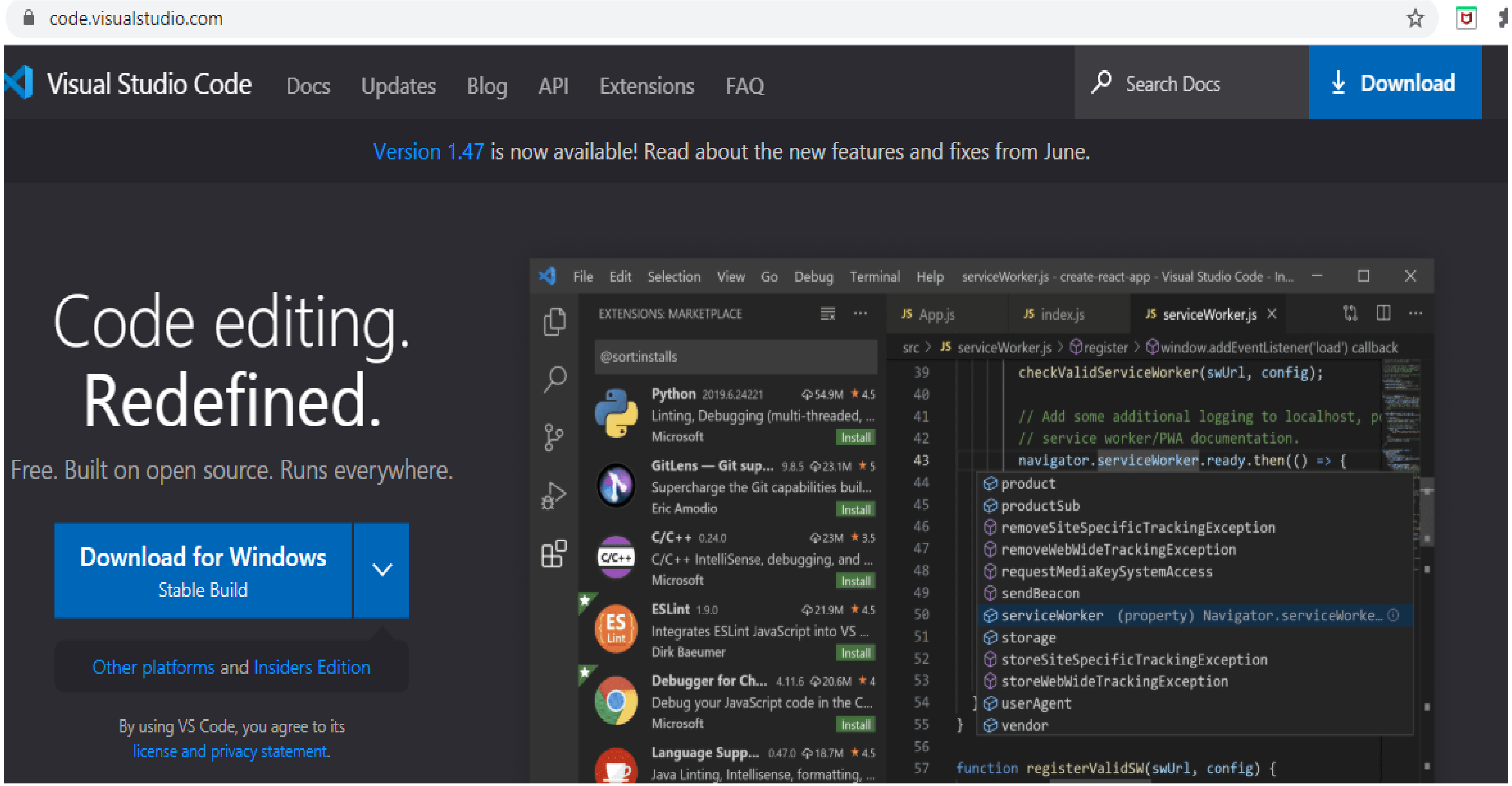
"This" variable

- **Regular function call**: the `this` keyword points at the global object, (the `window` object, in the browser).
- **Method call**: the `this` variable points to the object that is calling the method.
- *The `this` keyword is not assigned a value until a function where it is defined is actually called.*

# Programs Needed for Course

- **Visual Studio Code (Editor which I will use)**

https://code.visualstudio.com/


code.visualstudio.com
Visual Studio Code
Docs
Updates
Blog
API
Extensions
FAQ
Search Docs
Download
Download for Windows
Stable Build
Other platforms and Insiders Edition
File Edit Selection View Go Debug Terminal Help serviceWorker.js - create-react-app - Visual Studio Code - In...
EXTENSIONS: MARKETPLACE
@sort:installs
Python 2019.6.24221 54.9M 4.5
Linting, Debugging (multi-threaded, ...
Microsoft
Install
GitLens — Git sup... 9.8.5 23.1M 5
Supercharge the Git capabilities buil...
Eric Amodio
Install
C/C++ 0.24.0 23M 3.5
C/C++ IntelliSense, debugging, and ...
Microsoft
Install
ESLint 1.9.0 21.9M 4.5
Integrates ESLint JavaScript into VS ...
Dirk Baeumer
Install
Debugger for Ch... 4.11.6 20.6M 4
Debug your JavaScript code in the C...
Microsoft
Install
Language Supp... 0.47.0 18.7M 4.5
Java Linting, Intellisense, formatting, ...
App.js
index.js
serviceWorker.js
src > serviceWorker.js > register > window.addEventListener('load') callback
product
productSub
removeSiteSpecificTrackingException
removeWebWideTrackingException
requestMediaKeySystemAccess
sendBeacon
serviceWorker (property) Navigator.serviceWorke...
storage
storeSiteSpecificTrackingException
storeWebWideTrackingException
userAgent
vendor

Version 1.47 is now available! Read about the new features and fixes from June.

Code editing.
Redefined.

Free. Built on open source. Runs everywhere.

By using VS Code, you agree to its license and privacy statement.

```
checkValidServiceWorker(swUrl, config);

// Add some additional logging to localhost, p
// service worker/PWA documentation.
navigator.serviceWorker.ready.then(() => {
```

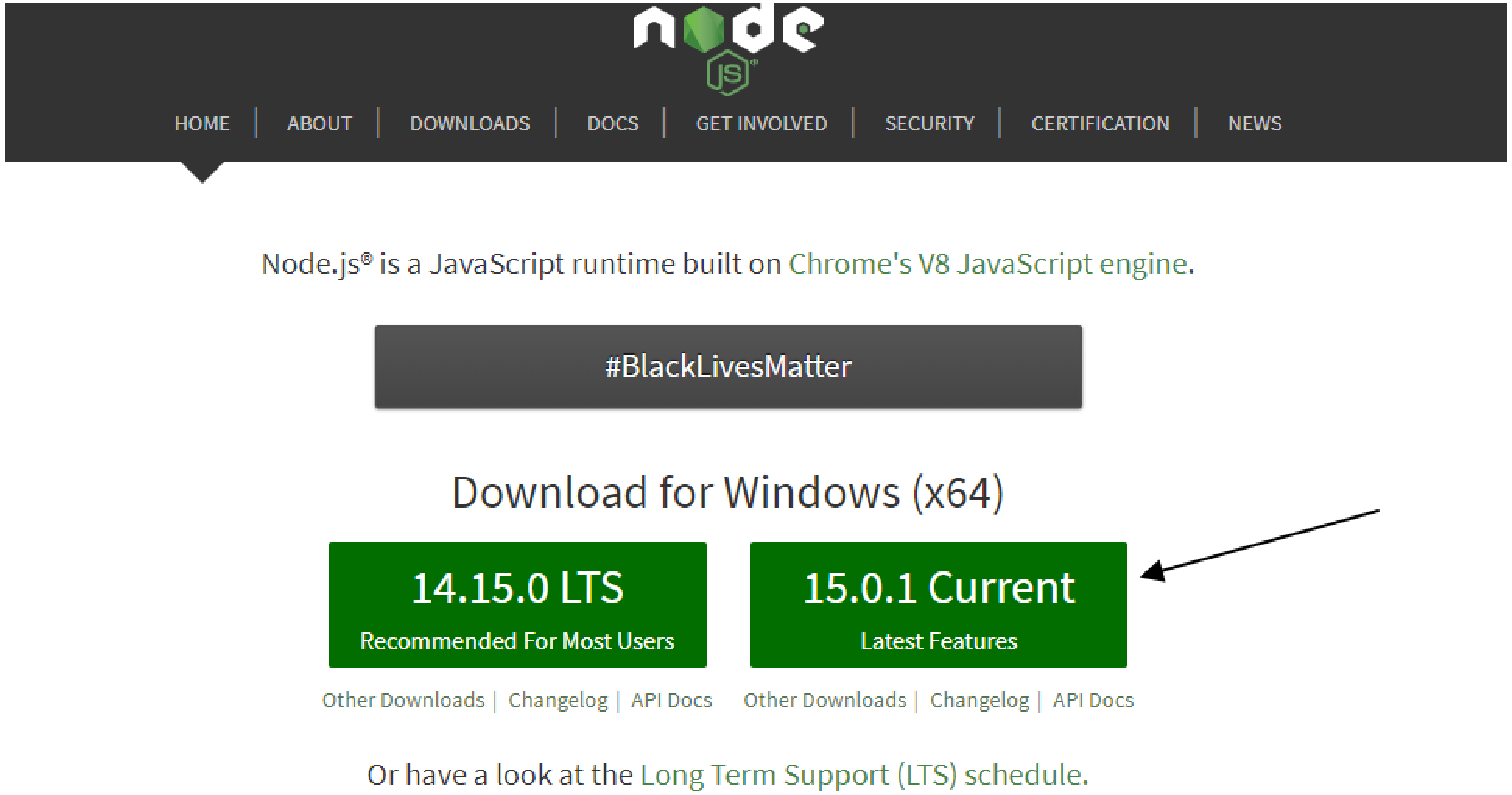
```
    }
  }

function registerValidSW(swUrl, config) {
```

# Programs Needed for Course

- **Node Js (**JavaScript runtime – server side**)**

[https://nodejs.org/en/](https://nodejs.org/en/)


HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | CERTIFICATION | NEWS

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

Download for Windows (x64)

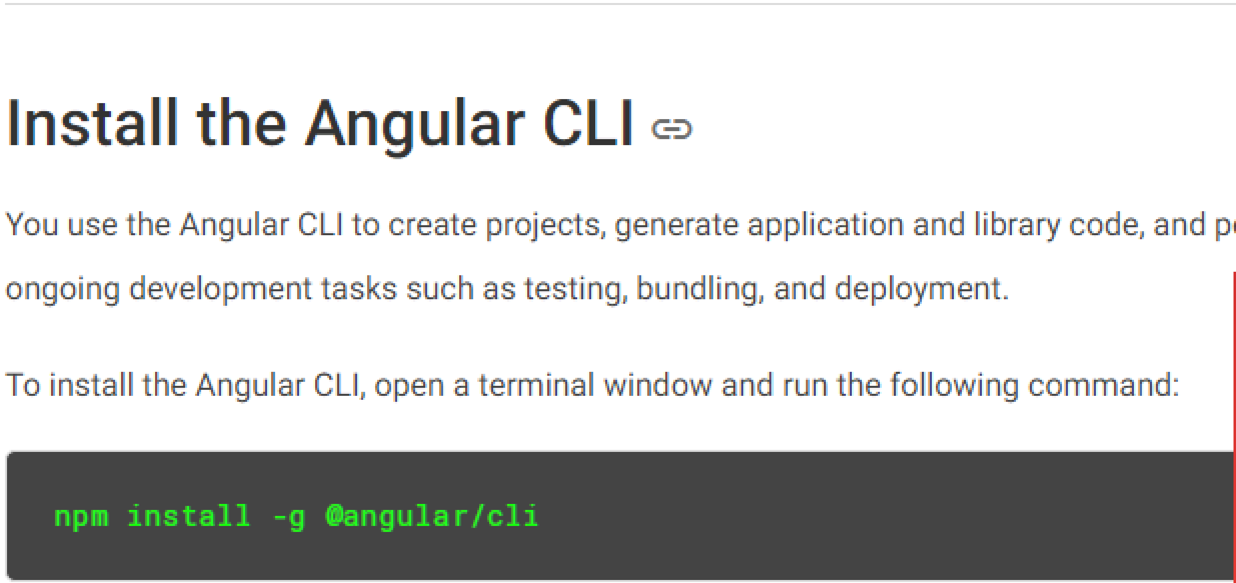
14.15.0 LTS
Recommended For Most Users
15.0.1 Current
Latest Features
Other Downloads | Changelog | API Docs
Other Downloads | Changelog | API Docs

Or have a look at the Long Term Support (LTS) schedule.

# Programs Needed for Course

- **Angular (CLI)**

[https://angular.io/cli](https://angular.io/cli)

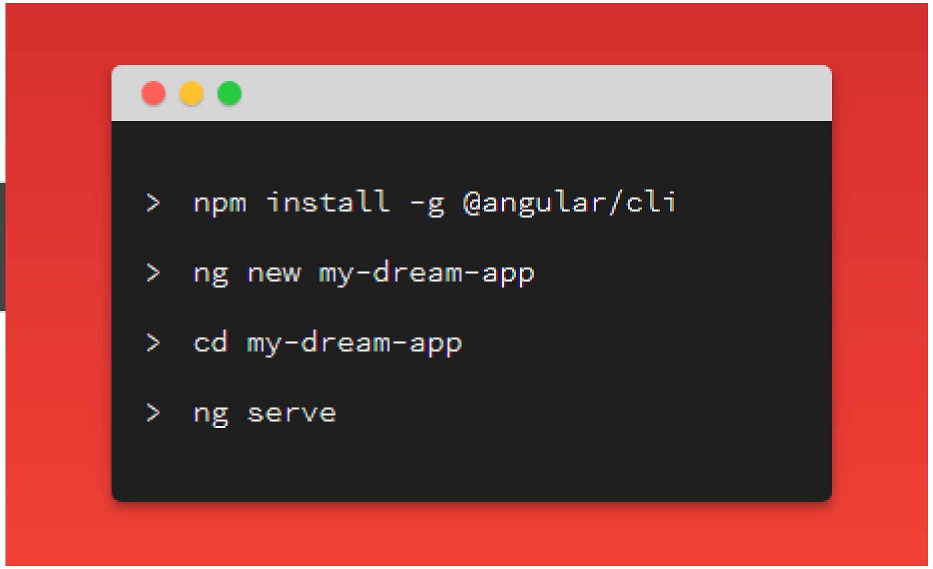


## Install the Angular CLI

You use the Angular CLI to create projects, generate application and library code, and perform a variety of ongoing development tasks such as testing, bundling, and deployment.

To install the Angular CLI, open a terminal window and run the following command:

```
npm install -g @angular/cli
```



```
> npm install -g @angular/cli
> ng new my-dream-app
> cd my-dream-app
> ng serve
```

# Thank You