SIQI (FERMI) FEI

CSC309 Week 2

GIT & JAVASCRIPTS & DOM

HTTPS://WWW.CS.TORONTO.EDU/~KIANOOSH/COURSES/CSC309H5/

10 JANUARY, 2025

Please join the Zoom for polls

Meeting Code:

Password:

Breakdown

Welcome to second tutorial of CSC309!

GIT

- BASIC CONCEPTS
- BRANCHING
- IN THE REAL WORLD & WORK AS A TEAM

JAVASCRIPTS

- INTRO
- SCRIPTS
- OBJECTS & PROPERTIES
- EVENTS & METHODS
- THIS
- FUNCTIONS

DOM

- INTRO

GIT

Basic Concepts

WHAT IS GIT?

Git is a distributed version control system that helps developers track changes in their codebase, collaborate on projects, and maintain a history of revisions. It is widely used in software development to streamline teamwork and manage project versions efficiently.

TO SEE MORE

HTTPS://GITHUB.COM/CSC207-UOFT/207-COURSE-NOTES/BLOB/MASTER/00-INTRODUCTION-TO-GIT.MD

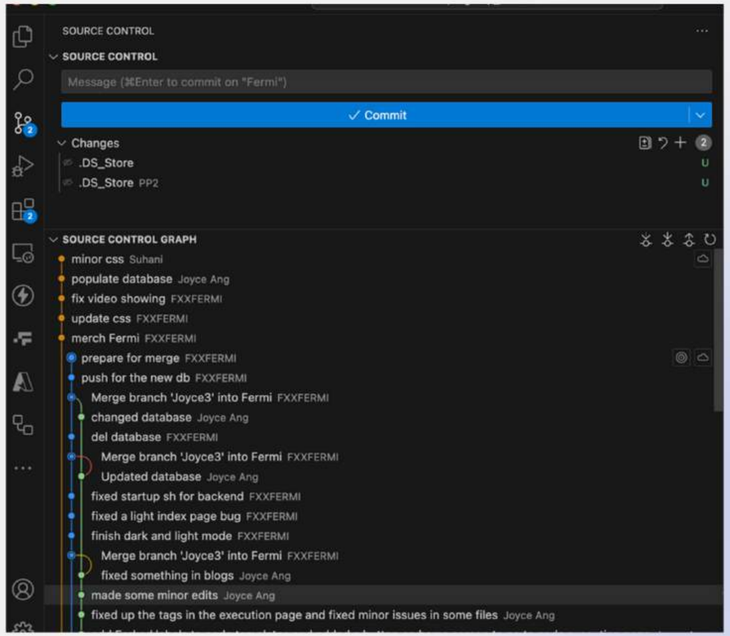
git ≠ GitHub

a tool/system

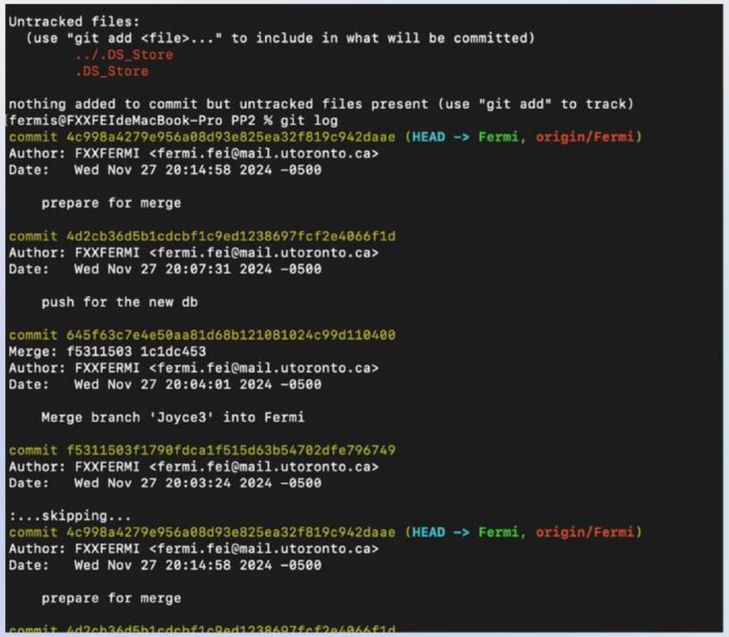
a service/platform

GIT IS INDEPENDENT OF GITHUB

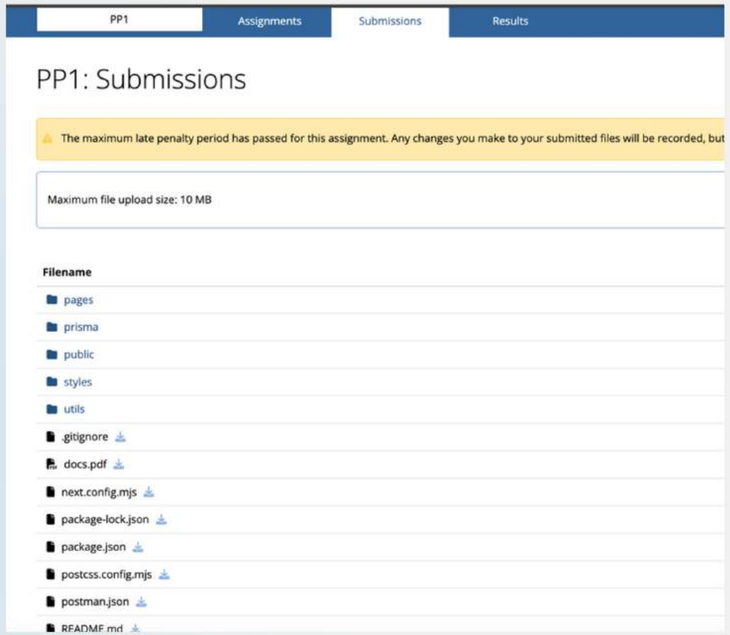
GIT



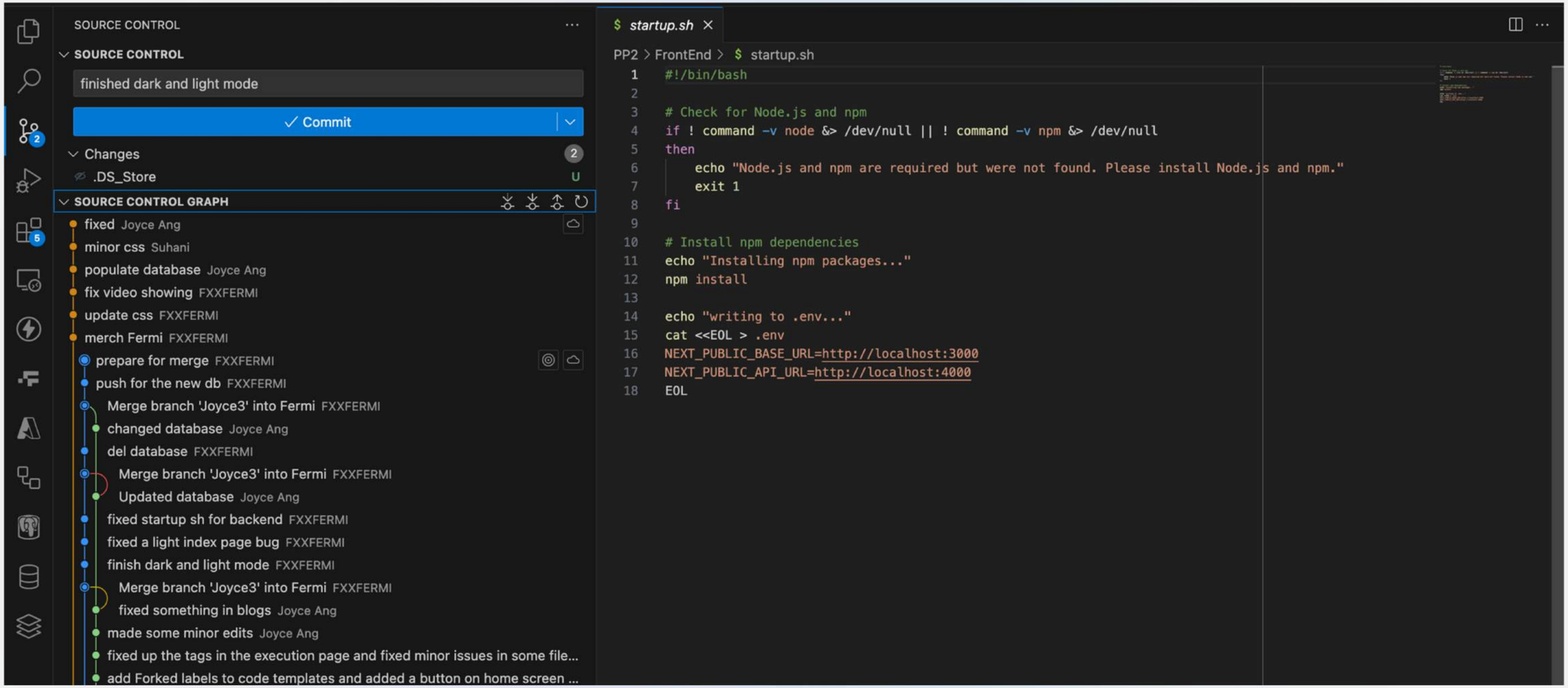
vscode



terminal

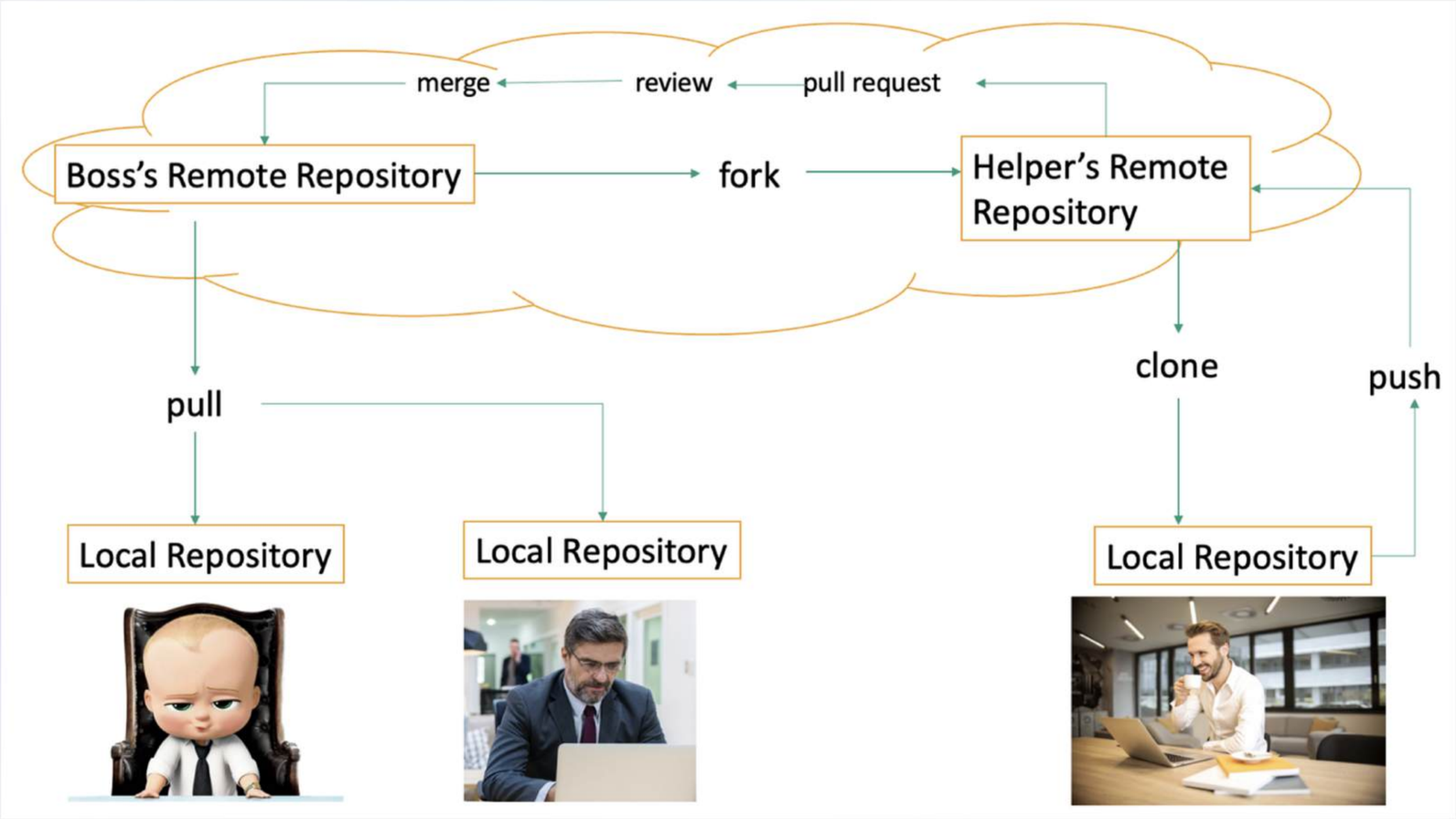


markus


SOURCE CONTROL
SOURCE CONTROL
finished dark and light mode
Commit
Changes
.DS_Store
SOURCE CONTROL GRAPH
fixed Joyce Ang
minor css Suhani
populate database Joyce Ang
fix video showing FXXFERMI
update css FXXFERMI
merch Fermi FXXFERMI
prepare for merge FXXFERMI
push for the new db FXXFERMI
Merge branch 'Joyce3' into Fermi FXXFERMI
changed database Joyce Ang
del database FXXFERMI
Merge branch 'Joyce3' into Fermi FXXFERMI
Updated database Joyce Ang
fixed startup sh for backend FXXFERMI
fixed a light index page bug FXXFERMI
finish dark and light mode FXXFERMI
Merge branch 'Joyce3' into Fermi FXXFERMI
fixed something in blogs Joyce Ang
made some minor edits Joyce Ang
fixed up the tags in the execution page and fixed minor issues in some file...
add Forked labels to code templates and added a button on home screen ...
startup.sh
PP2 > FrontEnd > startup.sh
#!/bin/bash
Check for Node.js and npm
if ! command -v node &> /dev/null || ! command -v npm &> /dev/null
then
echo "Node.js and npm are required but were not found. Please install Node.js and npm."
exit 1
fi
Install npm dependencies
echo "Installing npm packages..."
npm install
echo "writing to .env..."
cat <<EOL > .env
NEXT_PUBLIC_BASE_URL=http://localhost:3000
NEXT_PUBLIC_API_URL=http://localhost:4000
EOL

DEMO

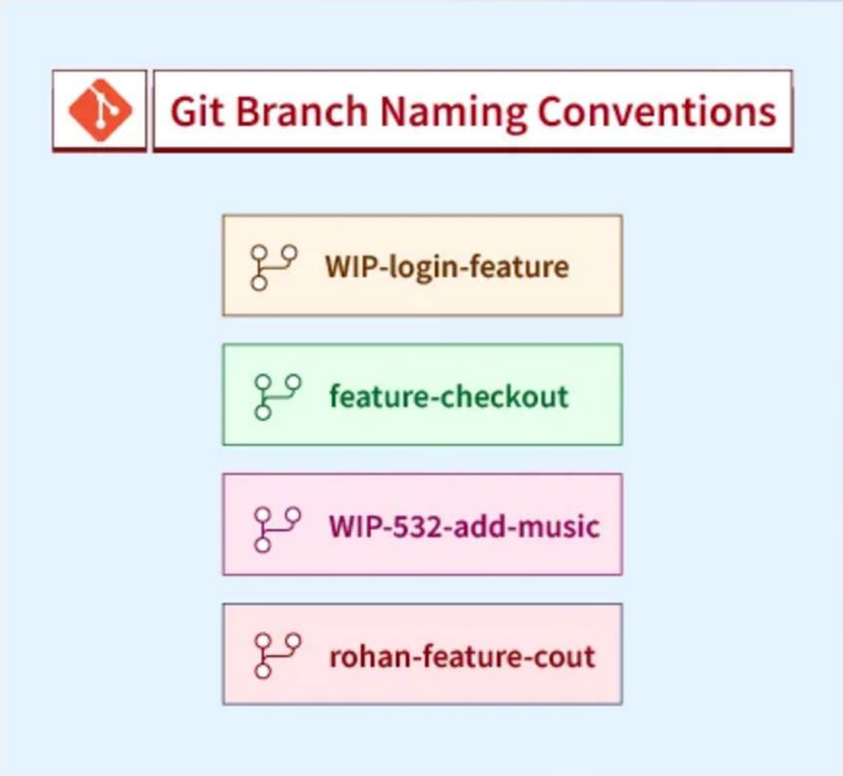
In the Real World & Work as a Team



Branching

Branching in Git is a powerful feature that allows developers to work on separate "branches" of a project simultaneously. It helps in isolating changes, experimenting, and managing features without affecting the main codebase.

```
fermis@FXXFEIdeMacBook-Pro group_10028 % git branch
* Fermi
  Joyce2
  Joyce3
  Suhani
```

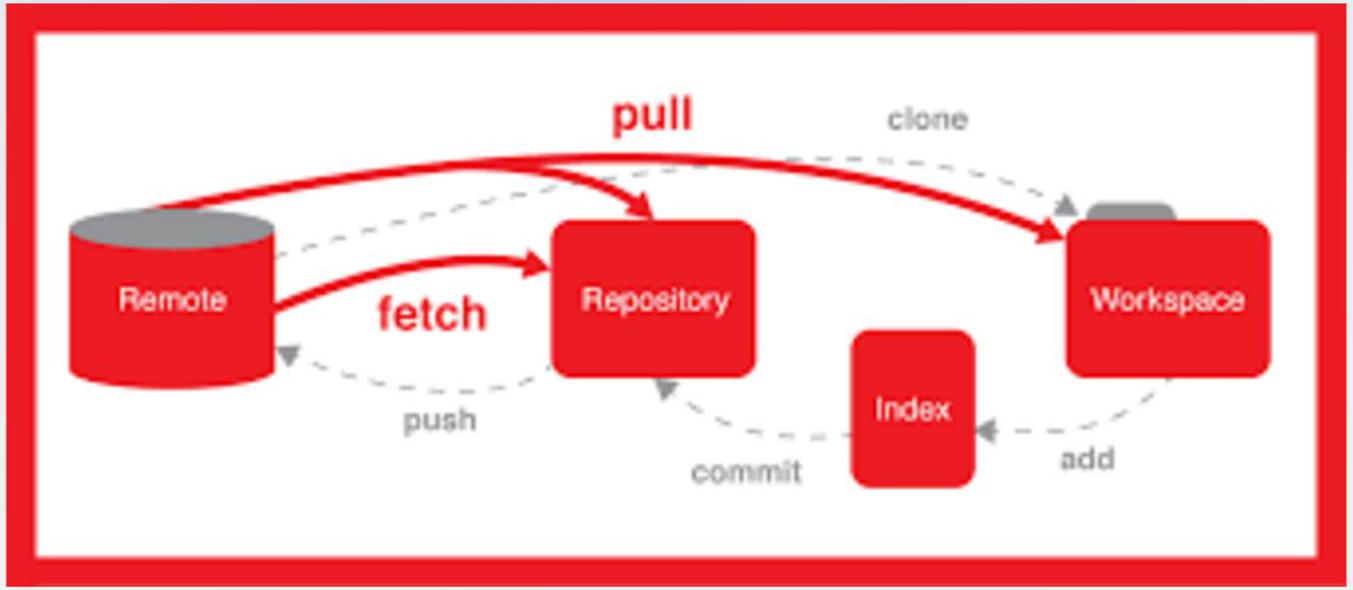


Branches

It is highly recommend to name your branches under your name for the course project.

Although in the real world business, people will name branches by the task/issue name.

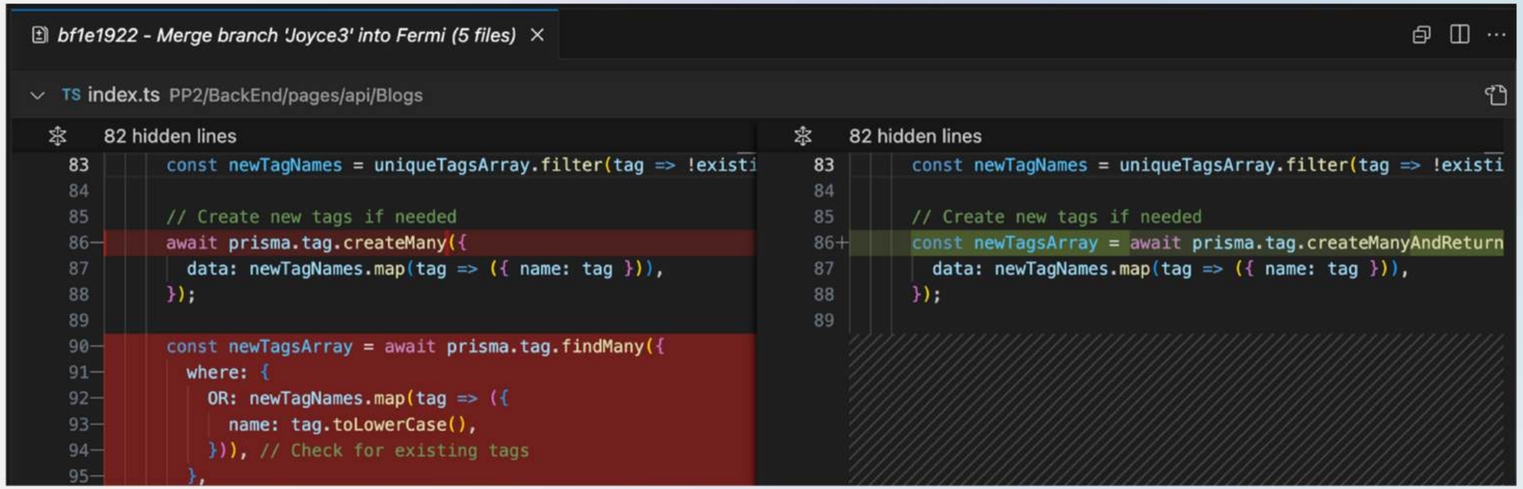
Please remeber
git fetch & git pull



Conflict

When you merge with conflicts, **DO NOT** rebase!!!!!! Try to solve it...

`**git blame**` can help to find who is responsible for the commit/conflict.



DEMO

```
13
14  /**
15   * Prints the welcome message
16  */
    Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
17  <<<<<<< HEAD (Current Change)
18  function printMessage(showUsage, message) {
19      console.log(message);
20
21  =======
22  function printMessage(showUsage, showVersion) {
23      console.log("Welcome To Line Counter");
24      if (showVersion) {
25          console.log("Version: 1.0.0");
26      }
27  >>>>>>> theirs (Incoming Change)
28      if (showUsage) {
29          console.log("Usage: node base.js <file1> <file2>  ... ");
30      }
31  }
32
33  /**
```

Resolve in Merge Editor

You, 20 seconds ago Ln 11, Col 26 Spaces: 4 UTF-8 CRLF {} JavaScript

```
XXFEIdeMacBook-Pro group_10028 % git commit -m"1
```

lol, idk what to say

✓ Commit

```
k-Pro group_10028 % git commit -m"fixed video playing bug"
```

finished dark and light mode

✓ Commit

Commit with useful messages

.gitignore

PP2 > FrontEnd > .gitignore

```
.env
.env.local
.env.development.local
.env.test.local
/prisma/migrations/

# dependencies
/node_modules
/.pnp
.pnp.*
.yarn/*
!.yarn/patches
!.yarn/plugins
!.yarn/releases
!.yarn/versions

# testing
/coverage

# next.js
/.next/
/out/

# production
/build

# misc
.DS_Store
```

Please Use .gitignore!!!!!!!

or .gitkeep

Add .gitignore

.gitignore template: **None**

Choose which files not to track from a list of templates. Learn more about ignoring files.

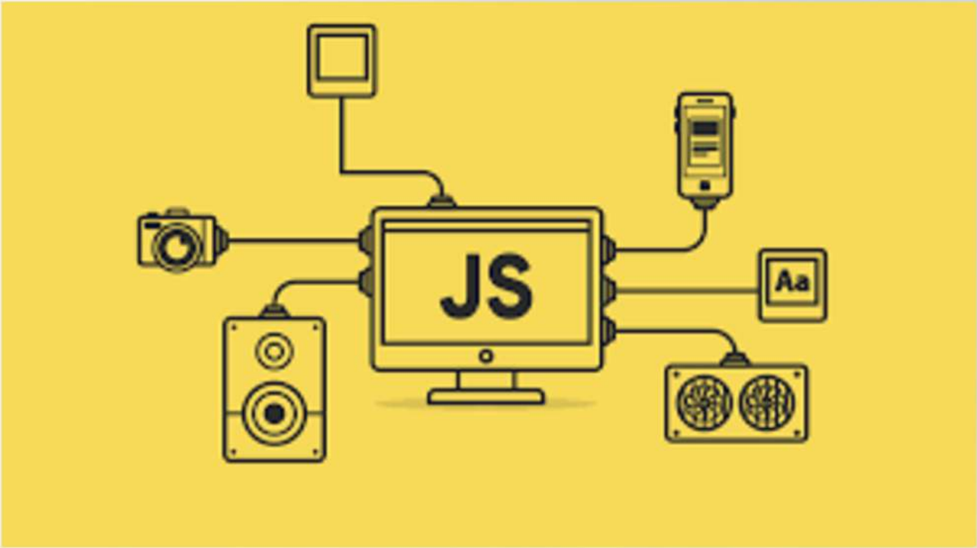
Remember to add **node_modules** to your .gitignore

For MacBook, remember add **.DS_Store**

```
Last login: Tue Jan  7 13:25:55 on ttys050
fermis@FXXFEIdeMacBook-Pro ~ % git clone ssh://                                                    /group_8327.git
```

DEMO

JavaScript



SCRIPTING LANGUAGE

It is a lightweight, interpreted, or just-in-time compiled language designed primarily for creating interactive and dynamic content on web pages. It is widely used for both client-side and server-side development.

JAVASCRIPT

```
19  const handleSubmit = async (e: React.FormEvent<HTMLFormElement>) => {
20    e.preventDefault();
21    try {
22      const response = await axios.post(
23        `${process.env.NEXT_PUBLIC_API_URL}/api/users/login`,
24        { username, password },
25        { withCredentials: true }
26      );
27
28      // Set the access token and refresh token in cookies
29      Cookies.set('accessToken', response.data.accessToken, { path: '/' });
30      // console.log(Cookies.get('accessToken'))
31      // Cookies.set('refreshToken', response.data.refreshToken, { path: '/' }
32
33      // Call the login function from AuthContext to set the global login state
34      login();
35
36      // console.log("Login successful:", response.data);
37
38      // Set the success message
39      setSuccessMessage("Login successful! Redi
40      setError(null);
41
42      // Delay the redirect by 3 seconds
43      setTimeout(() => {
44        router.push('/');
45      }, 3000);
46    } catch (error: any) {
47      // //console.error("Login failed:", error
48      setError(error.response?.data?.message ||
49      setSuccessMessage(null);
50    }
51  };
```

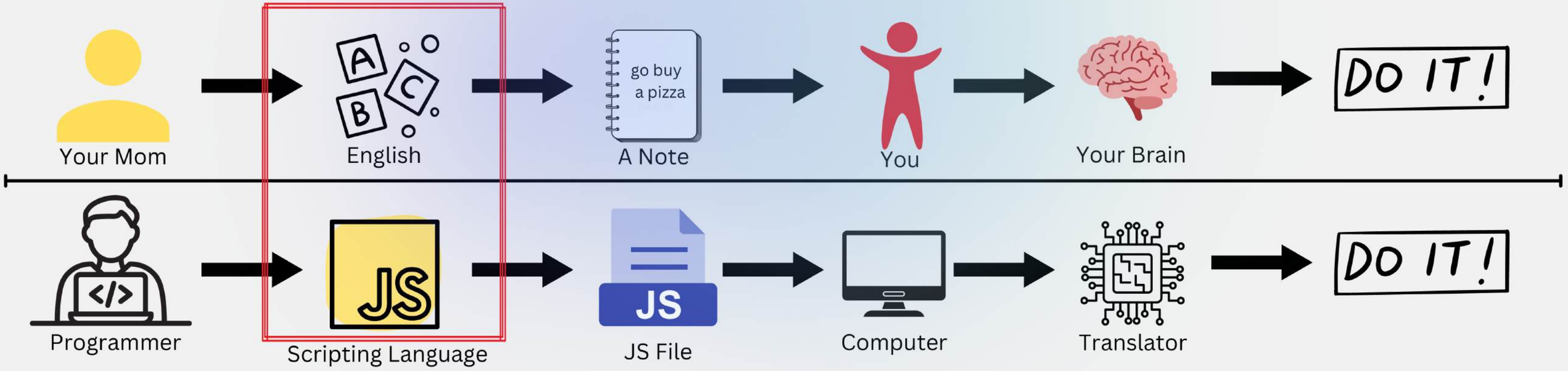
```
$ startup.sh ×
PP2 > FrontEnd > $ startup.sh
1   #!/bin/bash
2
3   # Check for Node.js and npm
4   if ! command -v node &> /dev/null || ! command -v npm &>
5   then
6       echo "Node.js and npm are required but were not foun
7       exit 1
8   fi
9
10  # Install npm dependencies
11  echo "Installing npm packages..."
12  npm install
13
14  echo "writing to .env..."
15  cat <<EOL > .env
16  NEXT_PUBLIC_BASE_URL=http://localhost:3000
17  NEXT_PUBLIC_API_URL=http://localhost:4000
18  EOL
```

Script

- When it comes to scripting vs programming (in the more general sense), you wouldn't use scripting or scripting languages to program static features like the overall appearance or layout of a website or web application, but you would use a scripting language to tell the static website to "do something," making your static content dynamic. -- *Scott Morris*

JAVASCRIPT

Scripting Language



SYSTEMS PROGRAMMING LANGUAGE

C JAVA ...

compiled rather than interpreted

SCRIPTING LANGUAGE

JS

directly tell the computer what to do

JAVASCRIPT

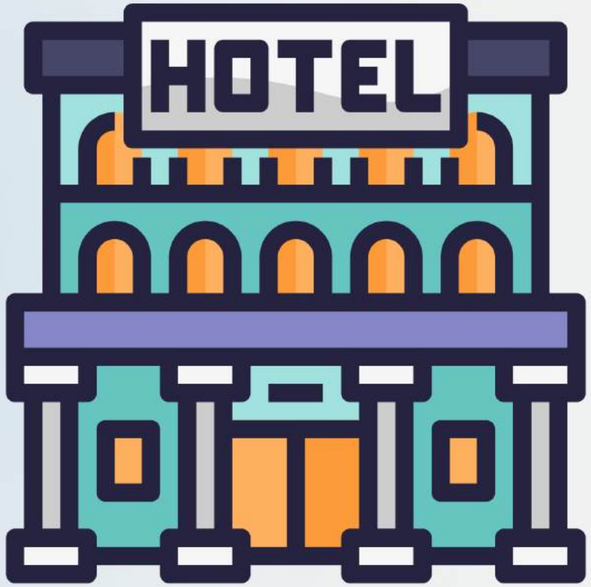
Embedded Language

Embedded Language → Embedded System

JAVASCRIPT

Objects & Properties

Object Type: Hotel



Object Type: Car

Object Type: Car

JAVASCRIPT

Object Type: Hotel

Properties:

name:	Hilton
rating:	4.1
rooms:	500
bookings:	210
gym:	false
pool:	true

Object Type: Car

Properties:

make:	Jeep
currentSpeed:	100km/h
color:	green
fuel:	gasoline

Objects => Things

Properties => Characteristics

Object Type: Hotel

Event	happens when:
book	reservation is made
Cancel	reservation is cancelled

Object Type: Car

Event	happens when:
brake	driver slow down
accelerate	driver speeds up

Events

In the real world, people interact with objects.
These interactions can change the values of the properties in objects.

Object Type: Hotel

Method	what it does:
makeBooking()	increase value of ***booking*** property
cancelBooking()	decrease value of ***booking*** property
checkAvailability()	subtracts value of ***bookings*** property from value of ***rooms*** property and returns number of rooms available

Object Type: Car

Method	what it does:
changeSpeed()	increase or decrease value of ***currentSpeed*** property

Methods

Methods represent things people need to do with objects.
They can retrieve or update the values of an object's properties.

Object Type: Hotel

Event	happend when:	method called:	properties	value
brake	driver slow down	changeSpeed()	make	Jeep
accelerate	driver speeds up	changeSpeed()	currentSpeed	10km/h
			color	500
			fuel	gasoline

Method	what it does:
changeSpeed()	increase or decrease value of currentSpeed property

Putting It All Together

Computers use data to create models of things in the real world.
The events, methods, and properties of an object all relate to each other:
Events can trigger methods, and methods can retrieve or update an object's properties.

JAVASCRIPT

Putting It All Together

Object Type: Hotel

Event	happend when:	method called:
book	reservation is made	makeBooking()
cancel	reservation is canceled	cancelBooking()

Method	what it does:
makeBooking()	increase value of ***booking*** property
cancelBooking()	decrease value of ***booking*** property
checkAvailability()	ubtracts value of ***bookings*** property from value of ***rooms*** property and returns number of rooms available

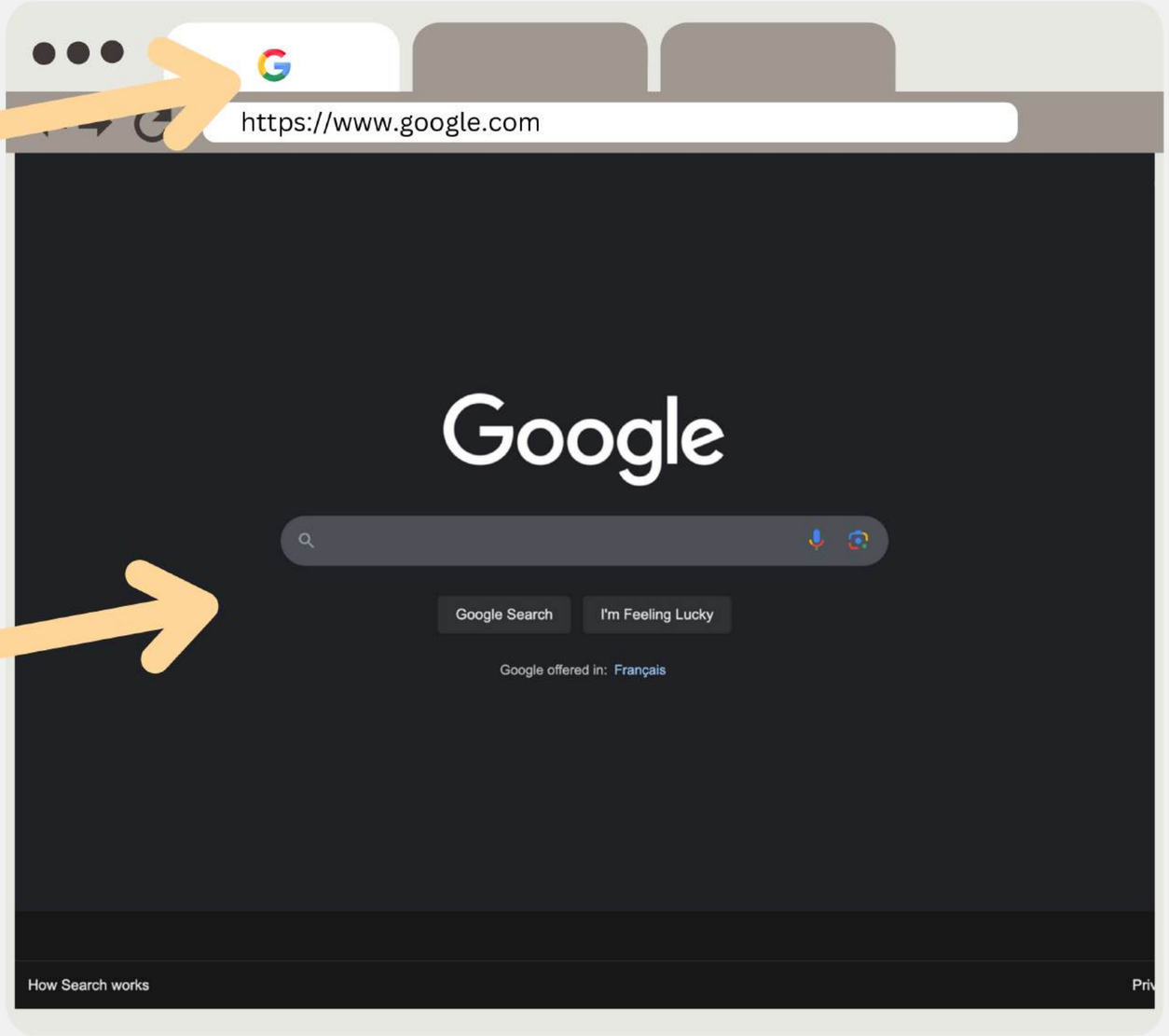
properties	value
name	Hilton
rating	4.1
rooms	500
bookings	210
gym	false
pool	true

JAVASCRIPT

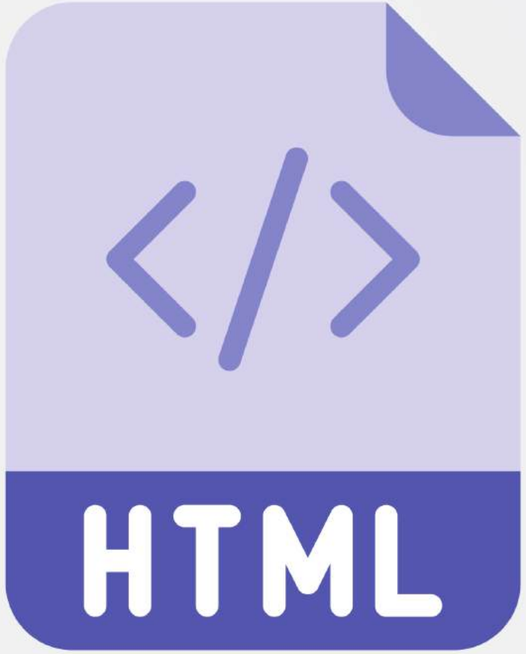
Look back to the browsers:

Object Type: Window	
Properties	**value**
location	https://www.google.com

Object Type: Document	
Properties	**value**
URL	https://www.google.com
lastModified	09/04/2014 15:33:37
title	google search bar



JAVASCRIPT

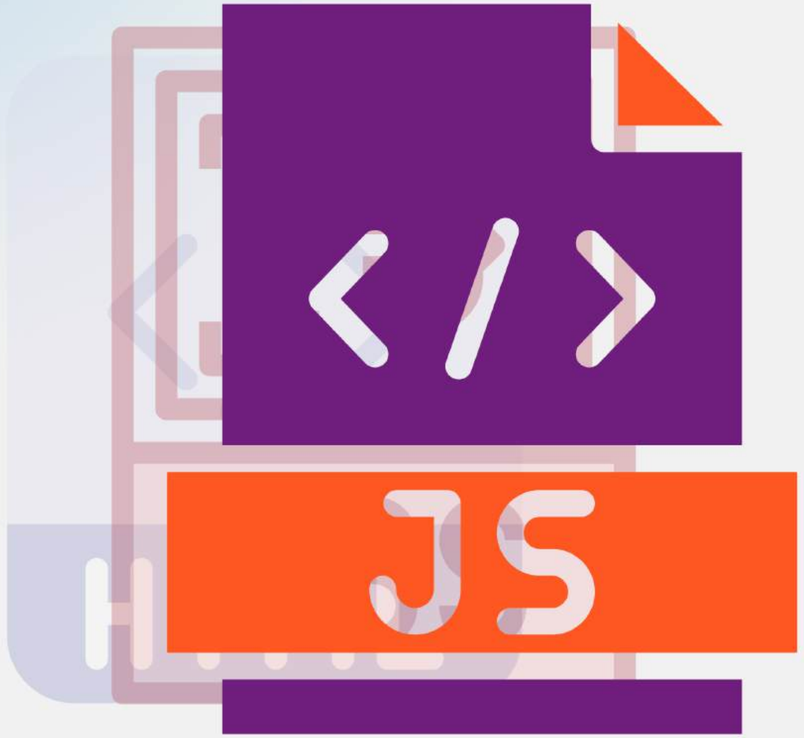


Content Layer

.html files

Presentation Layer

.css files



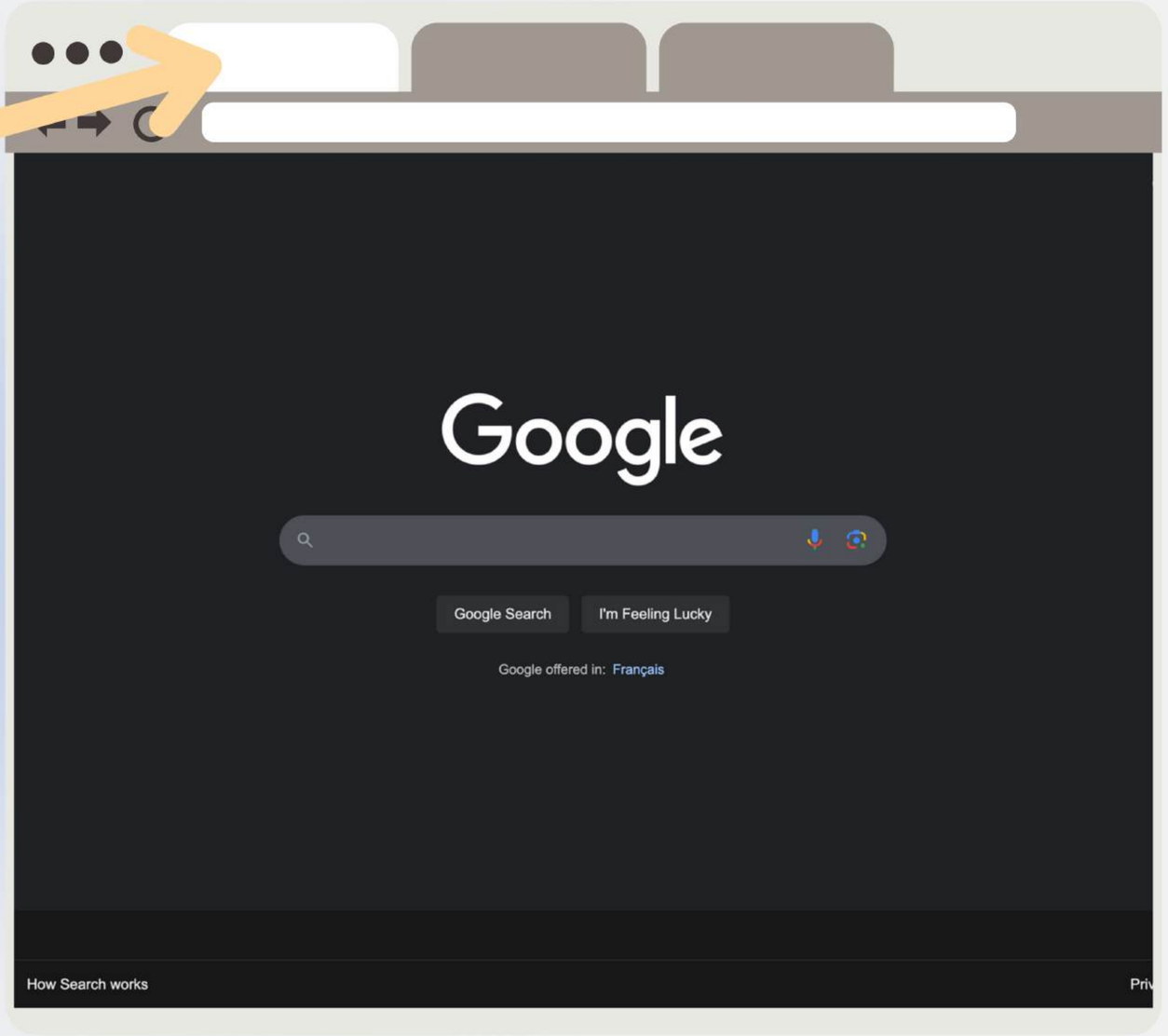
Behavior Layer

.js files

This

In JavaScript, this is a special keyword that refers to the execution context in which the current code is being executed. Its value depends on how and where the function is called, rather than where it is written.

Object Type: Window	
Properties	**value**
location	https://www.google.com



This

In the Global Scope

In non-strict mode, ***this*** in the global scope refers to the global object:

- In a browser, it's the window object.
- In Node.js, it's the global object.

In strict mode ("use strict";), this in the global scope is undefined.

JAVASCRIPT

```javascript
// In an Object Method
const person = {
    name: "Fermi",
    greet: function () {
      console.log(`Hello, my name is ${this.name}`);
    },
  };

  person.greet(); // Outputs: "Hello, my name is Fermi"
```

This

In an Object Method

When a method is called on an object, ***this*** refers to the object that the method belongs to.

JAVASCRIPT

```
// In a Regular Function
function showThis() {
    console.log(this);
  }

  showThis(); // Non-strict: `window` (browser) or `global` (Node.js)
  // Strict mode: `undefined`
```

This

In a Regular Function

In a regular function, ***this*** depends on how the function is called:

- In non-strict mode, it defaults to the global object (window or global).
- In strict mode, it defaults to undefined.

JAVASCRIPT

```javascript
//  In a Constructor Function or Class
class Person {
    constructor(name) {
      this.name = name;
    }
    greet() {
      console.log(`Hello, my name is ${this.name}`);
    }
  }

  const fermi = new Person("Fermi");
  fermi.greet(); // Outputs: "Hello, my name is Fermi"
```

This

In a Constructor Function or Class

When used in a constructor function or class, this refers to the new instance being created.

```javascript
//   In an Arrow Functions
const human = {
    name: "Fermi",
    greet: () => {
      console.log(this.name);
    },
  };

  human.greet(); // Outputs: `undefined` (Inherits `this` from global scope, not `human`)

```

This

In an Arrow Function

Arrow functions do not have their own this. Instead, they inherit this from the surrounding lexical scope.

```javascript
// In Event Handlers
const button = document.querySelector("button");
button.addEventListener("click", function () {
  console.log(this); // Refers to the button element
});
```

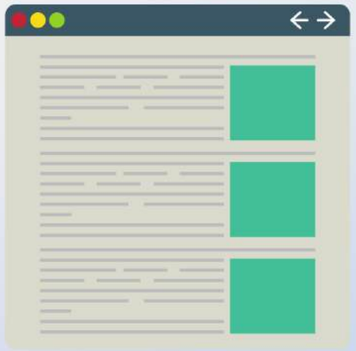
This

In Event Handlers

In an event handler, this typically refers to the element that triggered the event.

DOM

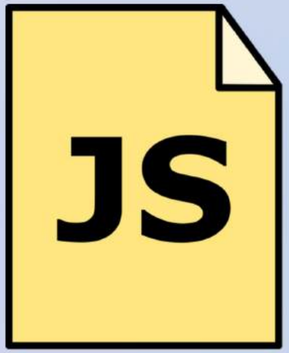
If:



A HTML Web Page

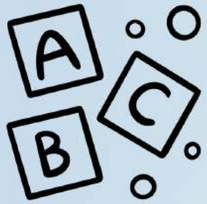
=

Arms and Legs



JavaScript

=



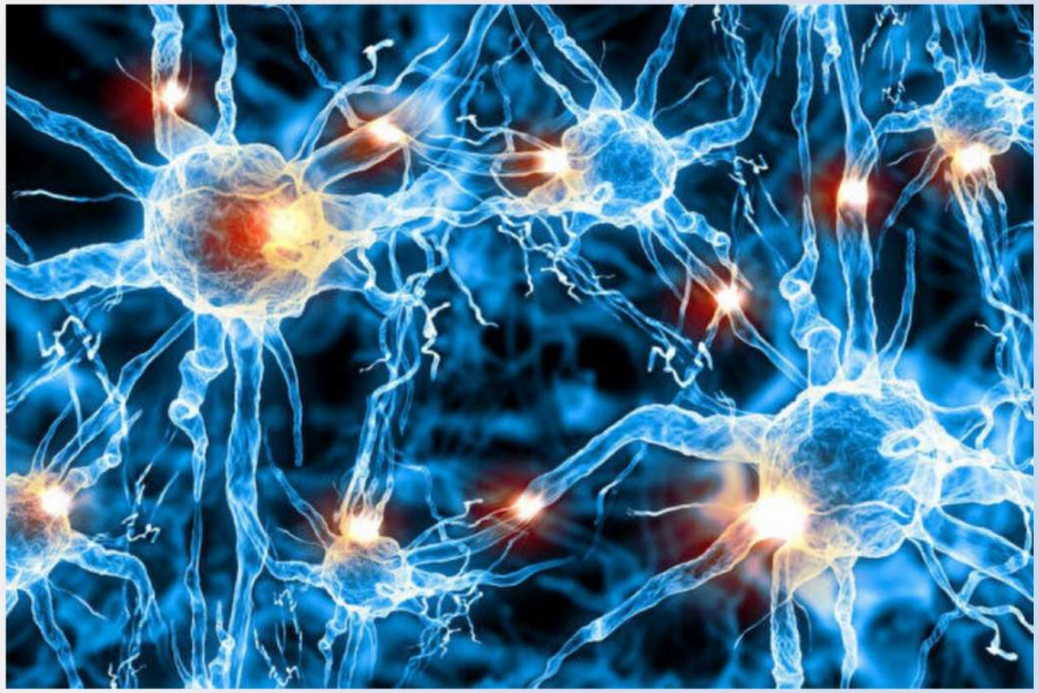
English

Browsers

=

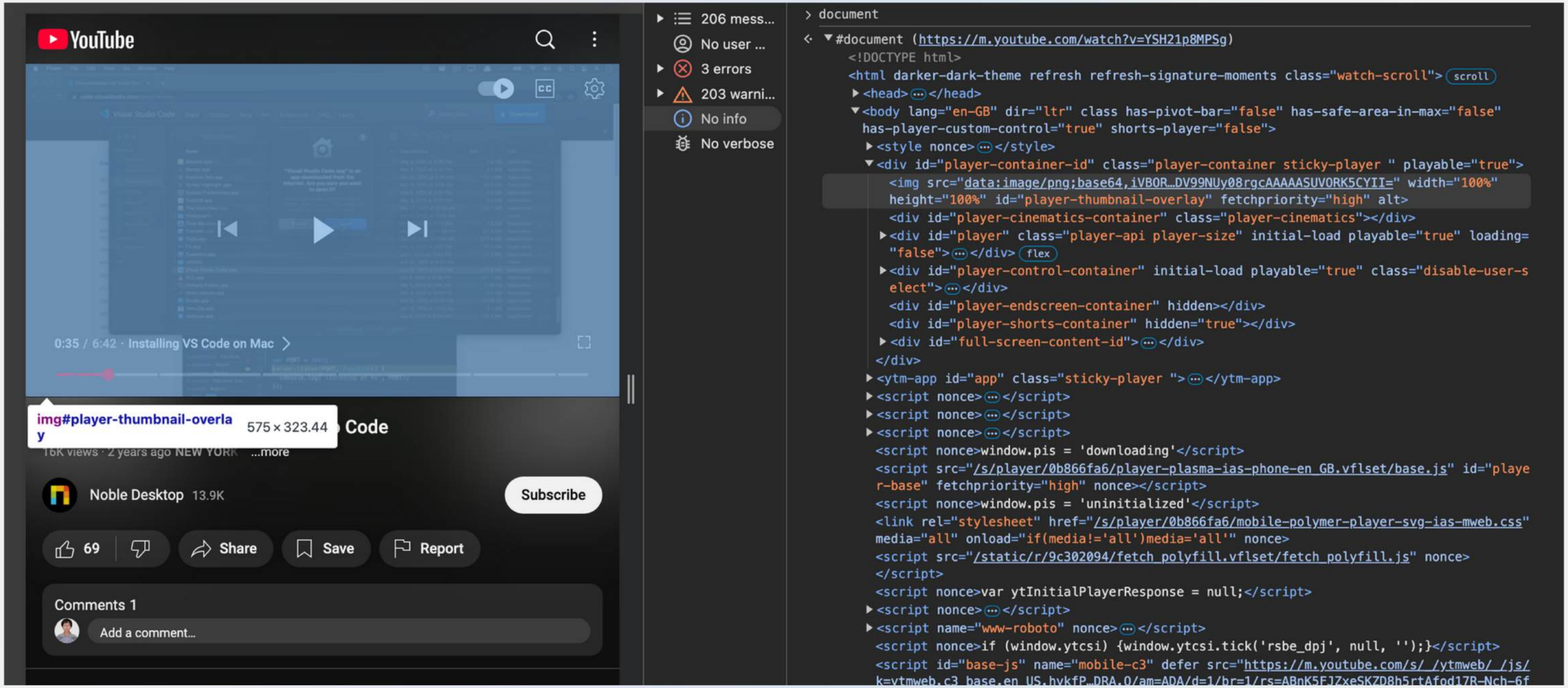
Brain

Document Object Model （DOM）



The Document Object Model (DOM) specifies how browsers should create a model of an HTML page and how JavaScript can access and update the contents of a web page while it is in the browser window.

In simpler terms: Think of it as the bridge between the HTML document and the JavaScript that interacts with it.


YouTube
0:35 / 6:42 · Installing VS Code on Mac
img#player-thumbnail-overlay
575 × 323.44
Code
16K views · 2 years ago NEW YORK ...more
Noble Desktop 13.9K
Subscribe
69
Share
Save
Report
Comments 1
Add a comment...
206 mess...
No user ...
3 errors
203 warni...
No info
No verbose
document
#document (https://m.youtube.com/watch?v=YSH21p8MPSg)
<!DOCTYPE html>
<html darker-dark-theme refresh refresh-signature-moments class="watch-scroll">
<head></head>
<body lang="en-GB" dir="ltr" class has-pivot-bar="false" has-safe-area-in-max="false" has-player-custom-control="true" shorts-player="false">
<style nonce></style>

<img src="…DV99NUy08rgcAAAAASUVORK5CYII=" width="100%" height="100%" id="player-thumbnail-overlay" fetchpriority="high" alt>

<div id="player" class="player-api player-size" initial-load playable="true" loading="false">
<div id="player-control-container" initial-load playable="true" class="disable-user-select">
<div id="player-endscreen-container" hidden>

<ytm-app id="app" class="sticky-player "></ytm-app>
<script nonce>window.pis = 'downloading'</script>
<script nonce>window.pis = 'uninitialized'</script>
<link rel="stylesheet" href="/s/player/0b866fa6/mobile-polymer-player-svg-ias-mweb.css" media="all" onload="if(media!='all')media='all'" nonce>
<script src="/static/r/9c302094/fetch_polyfill.vflset/fetch_polyfill.js" nonce>
<script nonce>var ytInitialPlayerResponse = null;</script>
<script name="www-roboto" nonce></script>
<script nonce>if (window.ytcsi) {window.ytcsi.tick('rsbe_dpj', null, '');}</script>
<script id="base-js" name="mobile-c3" defer src="https://m.youtube.com/s/_/ytmweb/_/js/

DEMO

How the DOM is Created

When a browser loads a webpage:

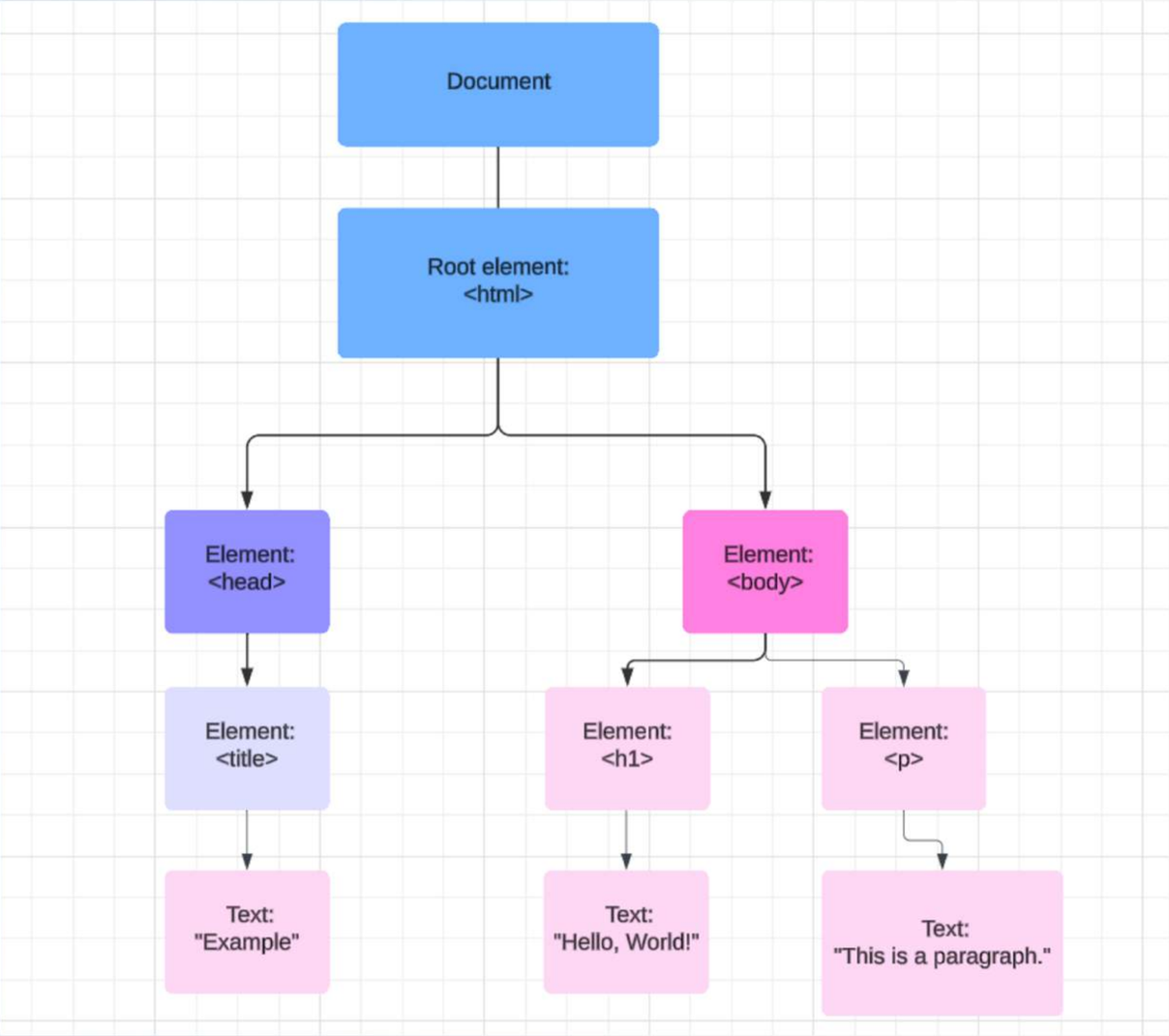
1. **HTML Parsing:**
    a. The browser reads the HTML file from the server or local file system.
    b. It parses the HTML to create the DOM tree structure.
2. **CSS Parsing:**
    a. The browser also parses CSS to create the CSS Object Model (CSSOM).
    b. The DOM and CSSOM combine to form the Render Tree, which the browser uses to display the webpage.
3. **Script Execution:**
    a. JavaScript can interact with the DOM using APIs to modify its structure, content, and styles.
    b. If a <script> tag is encountered during parsing, rendering may pause until the script is executed (unless it has the *async* or *defer* attribute).
4. **Rendering:**
    a. Once the DOM and CSSOM are constructed, the browser calculates the layout and paints the content to the screen.

DOM

tut.html U

tut.html > ...

```html
<!DOCTYPE html>
<html>
  <head>
    <title>Example</title>
  </head>
  <body>
    <h1>Hello, World!</h1>
    <p>This is a paragraph.</p>
  </body>
</html>
```



DEMO

SIQI (FERMI) FEI

Thank You

And my teammates, Joyce and Suhani, for their contributions to the group project.

CSC309 Week 2

10 JANUARY, 2025