DOM- DOCUMENT OBJECT MODEL

DOCUMENT OBJECT MODEL(DOM)

Create a folder: DOM-project, Add to VS Code, Create index.html

https://developer.mozilla.org/en-US/docs/Web/SVG/Attribute

Inline, internal, external css

1. Inline js add js to the html tag: This is not recommended

```
<body onload="alert('Hello World')">
```

2. Internal Javascript:use script tag in the body

Everything in the script can be javascript code

```
<script>alert('hello World')</script>
```

3. External Javascript file: by using src:

```
<script src="index.js"></script>
```

Where we put script tag is important

It should be on the last line

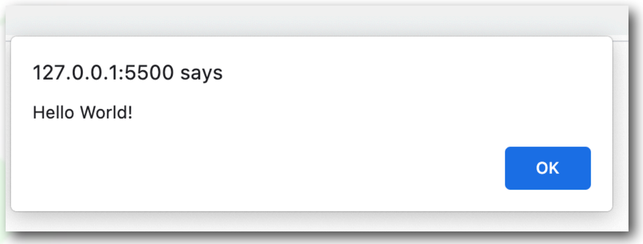
Cause first html, css should run, Then javascript should run

So use script in the body, in the last line.

For example, when you want to manipulate an element,But that is not there yet, Then we get error

INLINE

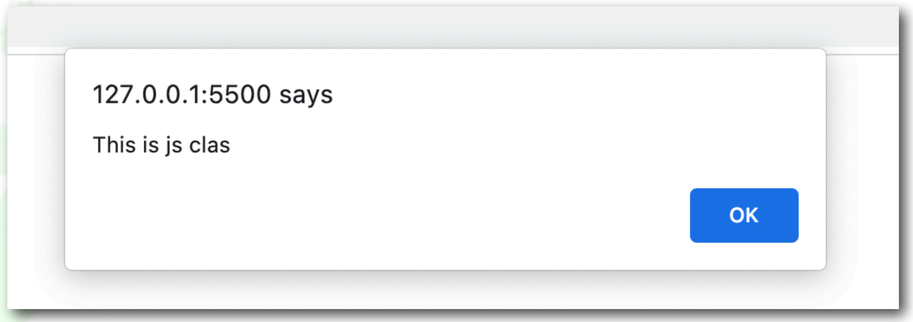
```html
<html>
    <head>
        <title>My Dom Project</title>
    </head>
    <body onload="alert('Hello World!')">
        <h1> My Heading </h1>
    </body>
</html>
```



INTERNAL

```html
<html>
    <head>
        <title>My Dom Project</title>
    </head>
    <body>
        <h1> My Heading </h1>
        <!-- script tag is used to write javascript codes
        This is internal javascript
    Note that we should use script tag at the end
Because, HTML should load first, then JS should run
-->
        <script>
            alert('This is js clas');
        </script>
    </body>

</html>
```

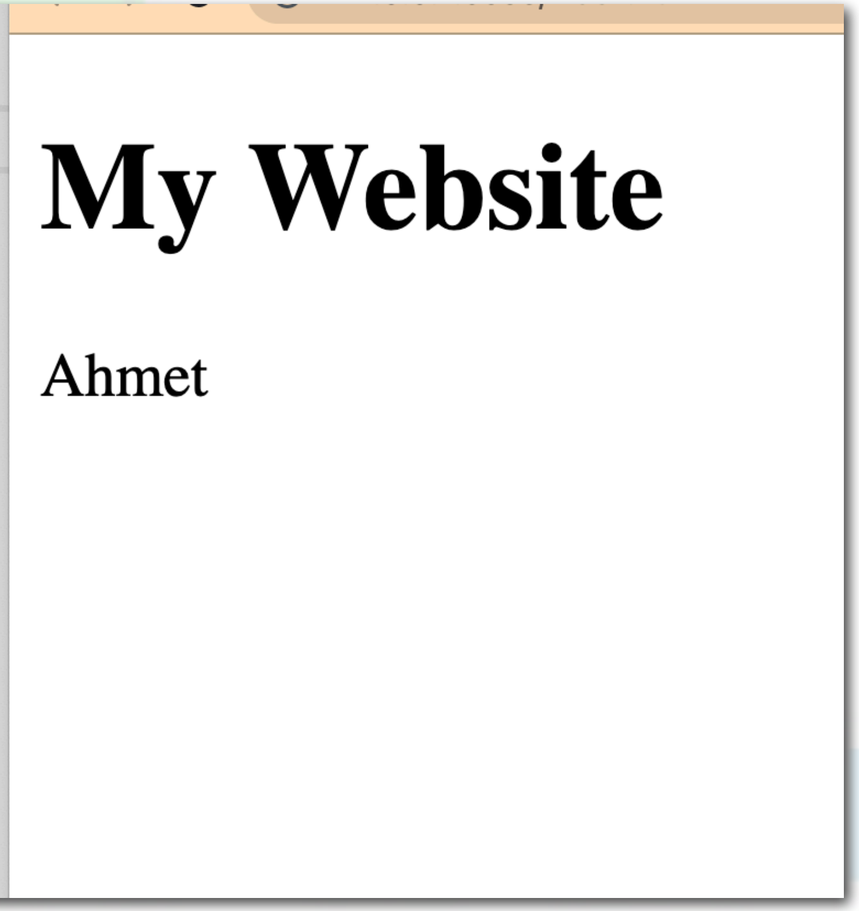


INTERNAL

WE CAN WITH ANY JS CODE INSIDE THE SCRIPT TAG
LIKE DECLARING VARIABLES AND ADDING THE NUMBERS,....

```
</head>
<body>
    <h1> My Website </h1>
    <script>
        var x=10; y=20
        console.log(x+y)
        console.log('Ahmet')
        // alert('Be Careful!!!')
        document.write('Ahmet')
    </script>

</body>
</html>
```



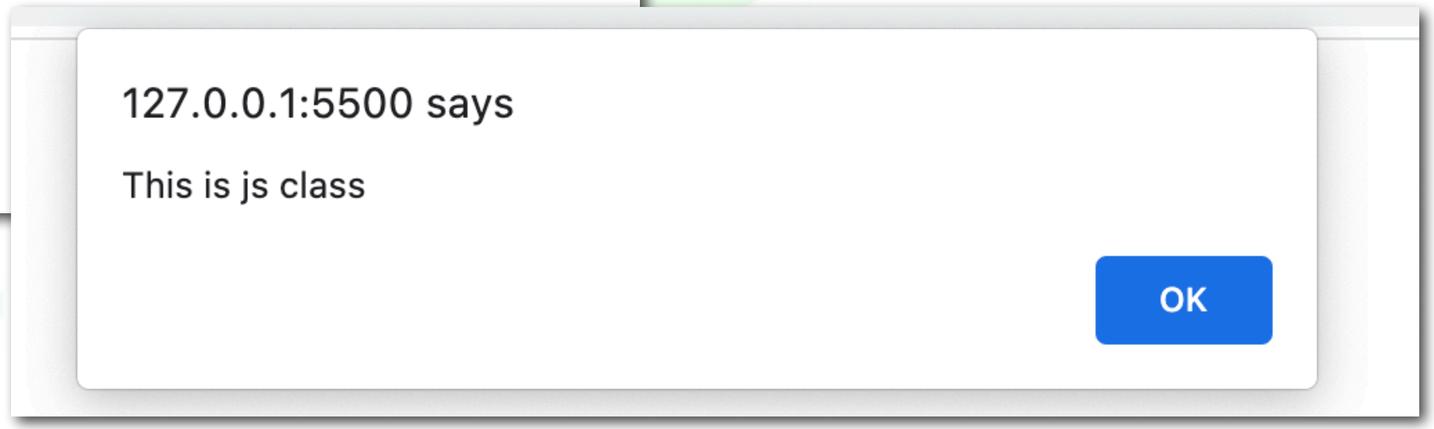
EXTERNAL JS

index.html > html > body > script

```
1   <html>
2       <head>
3           <title>My Dom Project</title>
4       </head>
5       <body>
6           <h1> My Heading </h1>
7   <!-- I am using index.js external file to write
8           <script src="index.js">
9
10          </script>
11      </body>
12
13  </html>
```

JS index.js

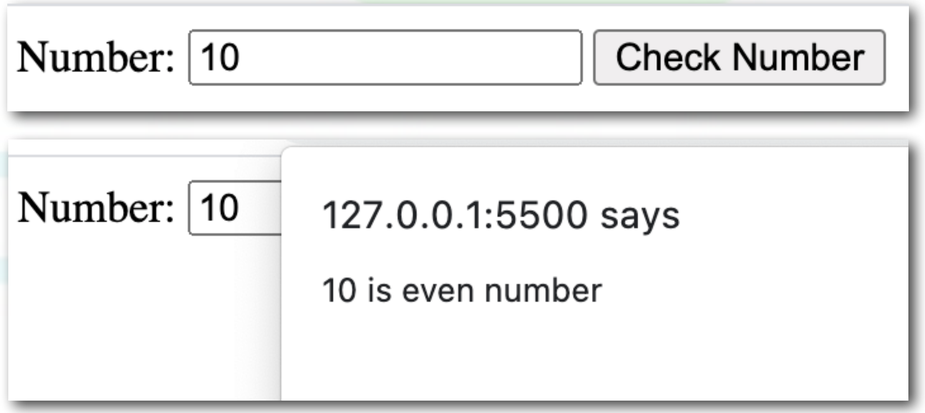
```
1   alert('This is js class');
```



TASK

Create a website
Website should have
Label : Number
Input
Button : CheckNumber

Functionality: Page should check if a number is even or odd
Alert Message : 10 is even number
Alert Message : 11 is odd number



NUMBER CHECKER SOLUTION

INTERNAL

```html
<html>
    <head>
        <title>My Dom Project</title>
    </head>
    <body>
      <form id="form">
          <label for="number label">Number : </label>
          <input type="text" id="TextBox">
          <input type="button" value="Check The Number"  onclick="IsEven()">
      </form>

    </body>
    <script>
        //Creating a function
        //We will use this function for the button element
        function IsEven(){
            // getting the value on the input element
            // when user enters 10, number=10
            // when user enters 29, number=29
            var number=document.getElementById("TextBox").value;
            if(number%2===0){
                alert(number + 'Number is Even');
            }else{
                alert(number + 'Number is Odd');
            }
        }
    </script>

</html>
```

EXTERNAL

index.html U JS index.js U

```js
//Creating a function
//We will use this function for the button element
function IsEven(){
    // getting the value on the input element
    // when user enters 10, number=10
    // when user enters 29, number=29
    var number=document.getElementById("TextBox").value;
    if(number%2===0){
        alert(number + ' Number is Even');
    }else{
        alert(number + ' Number is Odd');
    }
}
```

Welcome index.html U JS index.js U

index.html > html > script

```html
<html>
    <head>
        <title>My Dom Project</title>
    </head>
    <body>
      <form id="form">
          <label for="number label">Number : </label>
          <input type="text" id="TextBox">
          <input type="button" value="Check The Number"  onclick="IsEven()">
      </form>

    </body>
    <script src="index.js">

    </script>

</html>
```

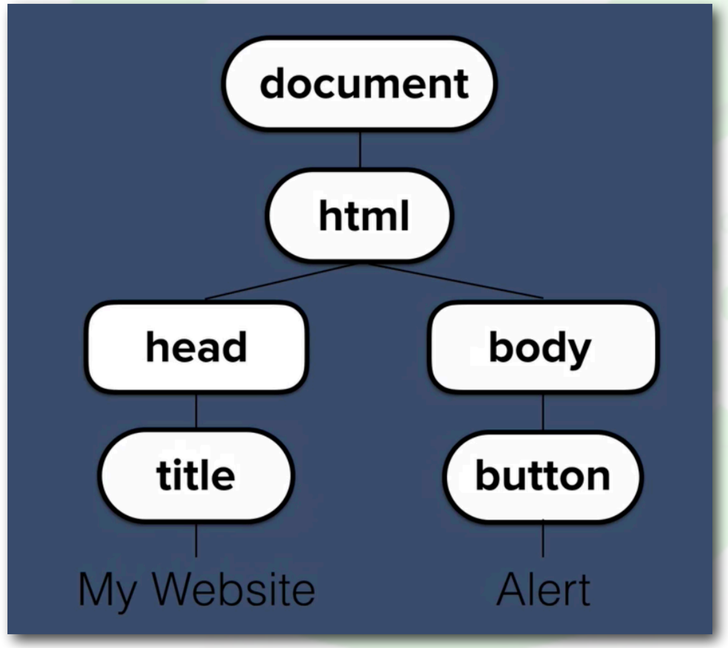
INTRO TO DOCUMENT OBJECT MODEL

Install html tree generator chrome extension

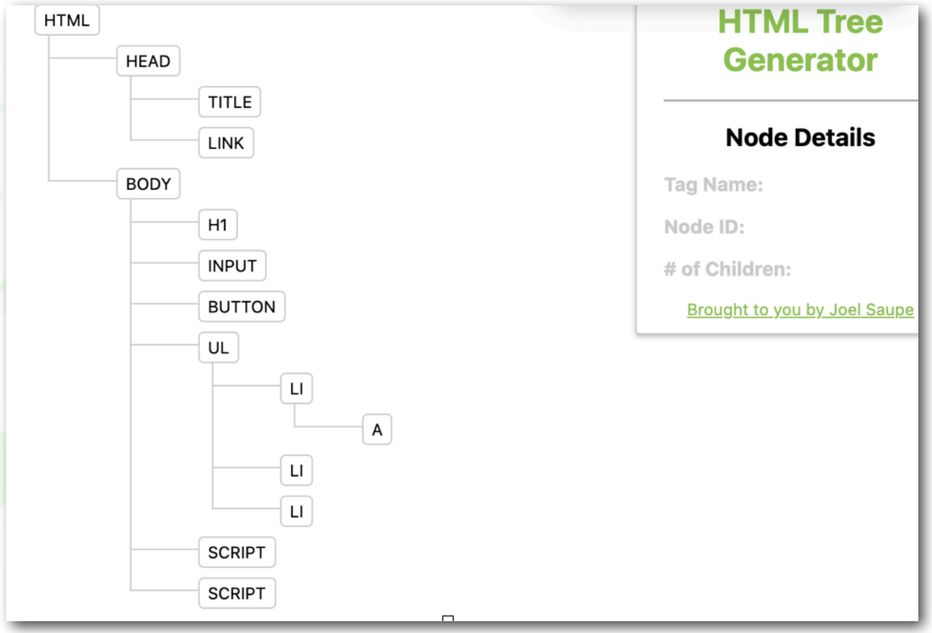
index.html

```
<html>
  <head>
    <title>My Website</title>
  </head>
  <body>
    <button>Alert</button>
  </body>
</html>
```

Dom tree



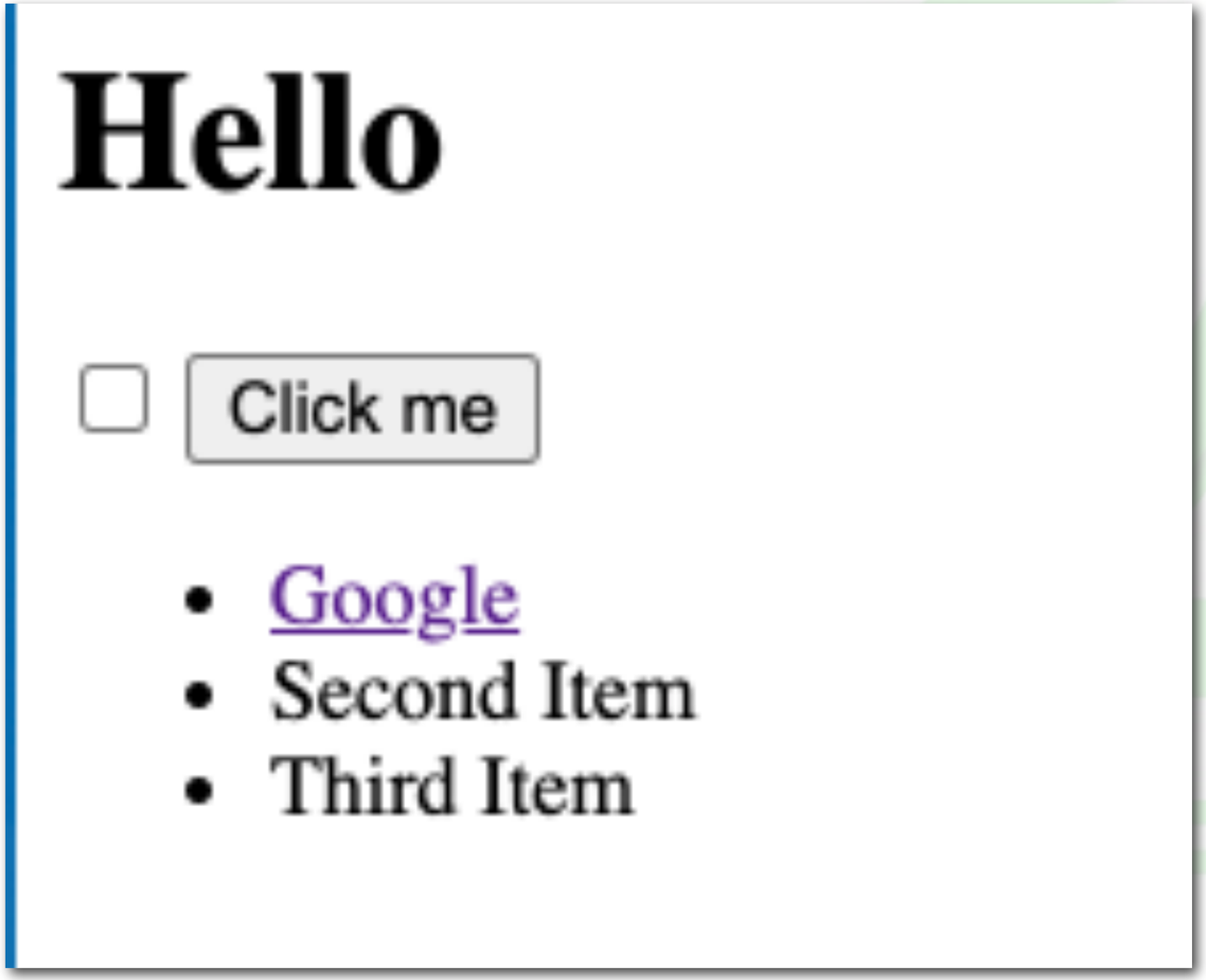
Html -dom -tree



CREATE A WORKSPACE

Crate a folder : dom-manipulation, Open in VS Code

Create index.html that shows below UI



INTRO TO DOCUMENT OBJECT MODEL

firstElementChild : Get the first child of the element

lastElementChild : Get the last child of the element

children[index] : Get the children at the given index

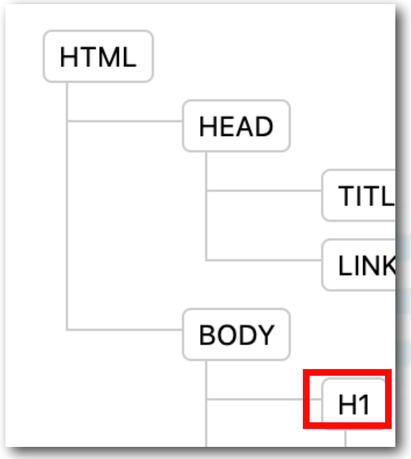
OPEN DEV CONSOLE AND START ACCESSING THE DOM OBJECTS

document.firstElementChild; —> html

document.firstElementChild.firstElementChild; —>head

document.firstElementChild.lastElementChild; —>body

HOW DO YOU SELECT h1 ELEMENT? html>body>h1



SOLUTION

HOW DO YOU SELECT h1 ELEMENT? html>body>h1

```
> document.firstElementChild;
<- <html>
     ▶ <head>…</head>
     ▶ <body>…</body>
   </html>
> document.firstElementChild.firstElementChild;
<- ▶ <head>…</head>
> document.firstElementChild.lastElementChild;
<- ▶ <body>…</body>
```

HOW DO YOU NAVIGATE THE SECOND ITEM IN THE LIST?

DOCUMENT.FIRSTELEMENTCHILD.LASTELEMENTCHILD.CHILDREN[3].CHILDREN[1]

```
> var secondItem =
  document.firstElementChild.lastElementChild.children[3].children[1]
<· undefined
> secondItem
<·  ▼<li>
      ::marker
      "Second Item"
    </li>
```

INTRO TO DOCUMENT OBJECT MODEL:**STYLE,CLICK(),INNERHTML**

We can manipulate elements using 2 ways: properties or methods

PROPERTIES : Describe something about the object

color
number of doors
engine power

We can manipulate using the js using . notation

car.color;//getting the value of the property

car.color=red;//setting a property

Properties:

To get and Set

- `innerHTML`
- `style`
- `firstChild`

METHODS. :Describes what objects can do

drive()
park()
accelerate()

car.drive();// calling the function

Methods:

Do actions

- `click()`
- `appendChild()`
- `setAttribute()`

TASK1: Change the color of the heading to red using the **style property**

TASK 2: Select checkbox and click on it. Use **click() method**

Task 3: Change the text of h1 as Welcome using **innerHTML property**

TASK SOLUTIONS

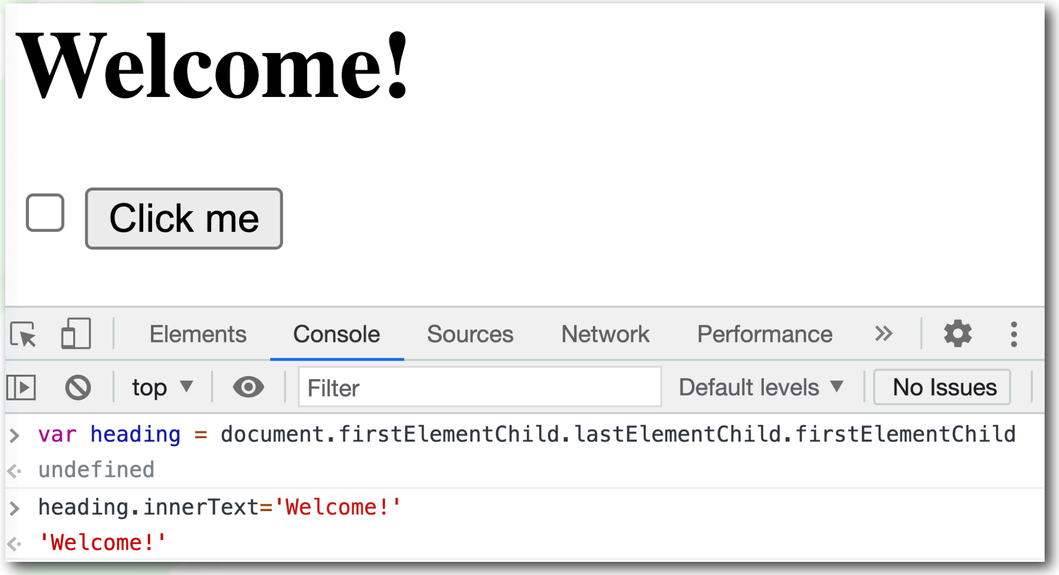
Task 1:Change the color of h1 as red

```
> var heading =
  document.firstElementChild.lastElementChild.firstElementChi
  ld
<· undefined
> heading.style.color='red'
<· 'red'
```

TASK 2: click on the checkbox

```
var
checkBox=document.firstElementChild.lastElementChild.chil
dren[1]
undefined
checkBox.click();
undefined
checkBox.click();
undefined
```

Task 3: Change the text of h1 as Welcome



INTRO TO DOCUMENT OBJECT MODEL:**INNERHTML,INNERTEXT,TEXTCONTENT**

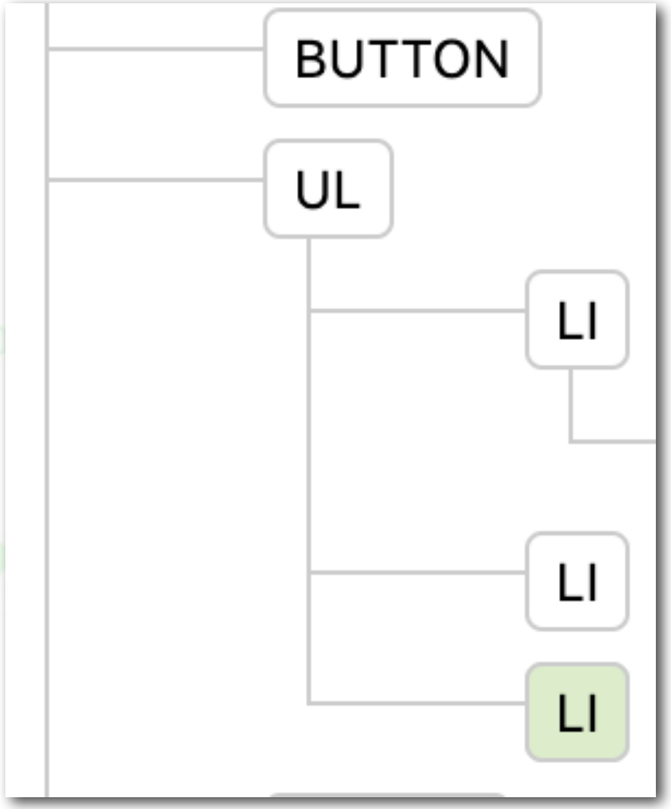
TASK 4: Select the 3rd li which is “Third Item”

and **change the text** as “Last Item”

You can only use dom without changing the html as we did

Use **innerHTML,innertext,textcontent** to

change get the text of the element



innerHTML
OR
innerText
&
children

SOLUTION

```
document.firstElementChild.lastElementChild.children[3].children[2].innerHTML='Last Item'
'Last Item'
```

TEXT MANIPULATION AND TEXT CONTENT PROPERTY

Difference is innerHTML gives the entire html
textContent give the texts only
Both changes the text that we select

We learned that we can change the text of elements using innerHTML

This means we can add inner html with new tag in the html on the fly: For example, we can change the text and make it italic using innerHTML

There is another class that does same thing, which is **textContent**

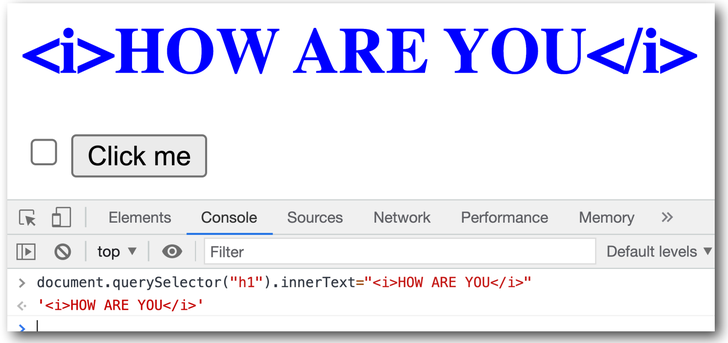
We can only add text using textContent

When I try to add a new tag, it will be sent as a text

TASK": Change the h1 to "HOW ARE YOU?" And make it italic.

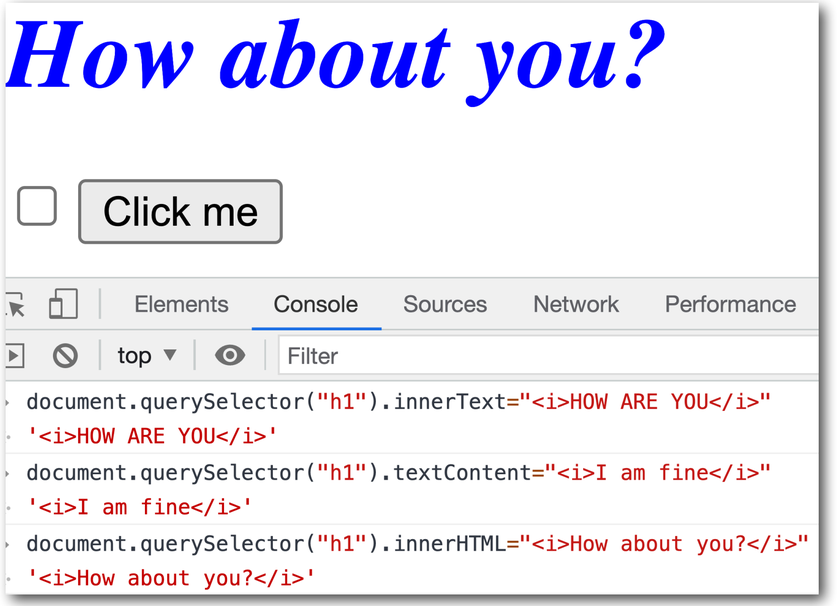
INNERTEXT VS TEXTCONTENT VS INNERHTML

innerText



textContent

InnerHTML



SELECTING HTML ELEMENTS WITH JAVASCRIPT

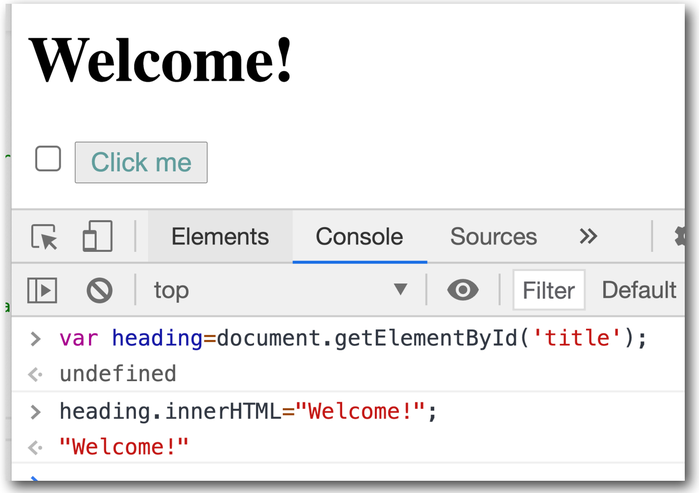
document.getElementById(“id name”);

—->selecting element based on the id name, which is unique, **returns SINGLE UNIQUE ELEMENT!!!**

getElementById(); is used to get the single item since id must be unique.

This returns **single item**-not array of item.

So we can get the element and manipulate directly



TASK: Change the text of the heading to “Welcome!”

document.getElementById(“title”);

SELECTING HTML ELEMENTS WITH JAVASCRIPT

document.getElementsByClassName(“class name”);

—->selecting element based on the class name, **returns an array** even if there is one item, **so use index to manipulate**

TASK: Change the button color to purple

```
<button class="btn">Click me</button> == $0
```

document.getElementsByClassName(“**btn**”);

Without index returns error

Pass index even though there is Unique array

```
Failed to load resource: the server responded with a status of 404 (Not Found)   favicon.ico:1
> document.getElementsByClassName("btn").style.color="red";
Uncaught TypeError: Cannot set property 'color' of undefined   VM9584:1
    at <anonymous>:1:51
> document.getElementsByClassName("btn")[0].style.color="purple";
< "purple"
```

SELECTING HTML ELEMENTS WITH JAVASCRIPT

document.getElementsByTagName("tag name");

—> search for element with specific tame, returns array

```
> document.getElementsByTagName("li");
<· ▼HTMLCollection(3) [li.item, li.item, li.item]
    ▶0: li.item
    ▶1: li.item
    ▶2: li.item
     length: 3
    ▶__proto__: HTMLCollection
```

document.getElementsByTagName("li"); => give me all li tags in the document

TASK: Change the color of the Third Item element to green

```
▼<ul>
  ▶<li>…</li>
  ▶<li>…</li>
  ▶<li>…</li> == $0
 </ul>
```

document.getElementsByTagName("li");

- Google
- Second Item
- Third Item

Elements | Console | Sources | »

top | Filter | Default levels

No Issues

```
> document.getElementsByTagName('li')[2].style.color='green'
< 'green'
```

*****SELECTING HTML ELEMENTS WITH JAVASCRIPT

```
document.querySelector(" multiple ways ");
```

——>**returns single element. When multiple element, returns first element**

Only tag name Or .className Or #id Or we can combine different things like tag and id, id and class etc

document.querySelector("**#title .container-fluid**"); **means id title and class container**

When we are using the combination of the elements, we must use space in between the elements.

TASK: Select h1, button, first li, anchor using querySelector

SOLUTIONS

Location h1 heading using querySelector

```
> document.querySelector('h1');
  ▶<h1 id="title" class="heading-title" style="color: blue;">…</h1>
> document.querySelector('#title')
  ▶<h1 id="title" class="heading-title" style="color: blue;">…</h1>
> document.querySelector('.heading-title')
  ▶<h1 id="title" class="heading-title" style="color: blue;">…</h1>
```

Location button using querySelector

```
> document.querySelector('button')
    <button class="btn">Click me</button>
> document.querySelector('.btn')
    <button class="btn">Click me</button>
```

Location first li using querySelector

```
> document.querySelector('a')
    <a href>Google</a>
> document.querySelector('#list li a')
    <a href>Google</a>
> document.querySelector("ul[id='list']")
  ▶<ul id="list">…</ul>
> document.querySelector("ul[id='list'] a")
a
```

CSS LOCATORS:

Css locators are used to locate the DOM elements

tagName[attribute='attribute value']

Location third li(whose tag = li, class- third-item) using querySelector

```
> document.querySelector("li.third-item")
  ▶<li class="third-item">…</li>
```

QUERYSELECTORALL(“MULTIPLE WAYS”)

```
document.querySelectorAll(“ multiple ways ”);
```

This return multiple elements

We must use index to get specific element

TASK: Select last li item and change the color to blue

```
> document.querySelectorAll("li")[2].style.color='blue'
< 'blue'
```

CSS LOCATOR

CSS LOCATORS:

Css locators are used to locate the DOM elements

tagName[attribute='attribute value']

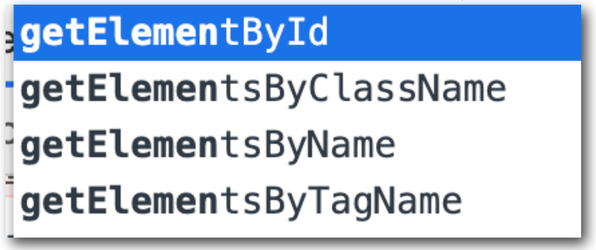
Location third li(whose tag = li, class- third-item) using querySelecto

```
> document.querySelector("li.third-item")
  ▶<li class="third-item">…</li>
```

WHICH SELECTOR SHOULD YOU USE?

->querySelector(); is a little more complex than selectById or selectByClassName but it is mostly used to reach specific element(id, class, element, tag name, and combine them)

->getElements is more broad and hard to get individual elements.



->getElementById can be the first choice since id is unique when element has an id

You can do **querySelector(“”); or querySelectorAll(“”);** because these two can do everything getElements methods does

After a while you will comfortable pick use best fitting selector

querySelector : returns one single element. I can’t use index. I can use style properly without index.

querySelectorAll : returns an array. I must use index. I cannot use style properly without index

MANIPULATING AND CHANGING STYLES OF HTML ELEMENTS WITH JS

DOM Styling(camelCase, Values passed as String) fontSize
CSS Styling(kebab-case, values are passed directly) font-size

Html styling and Javascript Stying is similar

style is camelCase fontSize and Values must be string.

https://www.w3schools.com/Jsref/dom_obj_style.asp

Dom object styles can be found here. This link give us how to use the javascript properties

TASK:

1.Change the heading color to red

2.Change the heading font-size(fontSize) 10rem

3.Change the heading padding 30%

SOLUTION

TASK:

1.Change the heading color to red

2.Change the heading font-size(fontSize) 10rem

3.Change the heading padding 30%

```
> document.querySelector("h1").style.color="red";
'red'
> document.querySelector("h1").style.fontSize="10rem";
'10rem'
> document.querySelector("h1").style.padding="30%"
'30%'
```

MANIPULATING AND CHANGING STYLES OF HTML ELEMENTS WITH JS

DOM Styling(camelCase, Values passed as String) -> fontSize
CSS Styling(kebab-case, values are passed directly) -> font-size

Use dev console to change do the tasks by css and javascript

TASK:

1. Change the click me button background color yellow

2. Hide the button

SOLUTION

TASK:

1. Change the click me button background color yellow

2. Hide the button

```
> document.querySelector("h1").style.backgroundColor="yellow"
'yellow'
> document.querySelector("h1").style.visibility="hidden"
'hidden'
```

JS STYLING VS CSS STYLING

CSS styling-kebab-case, value passed directly

DOM styling: camelCase, values are passed as string

```
document.querySelector("h1").style.fontSize="10rem";
'10rem'
```

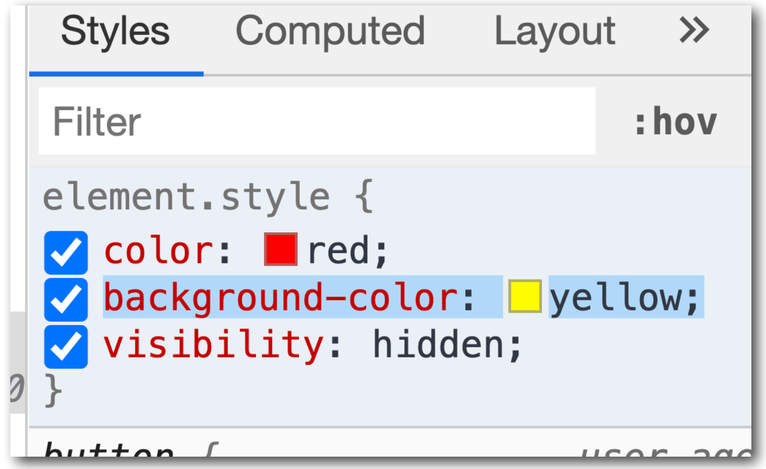
JS STYLING VS CSS STYLING

JAVASCRIPT STYLING
-CAMEL CASE
-STRING FOR ASSIGNMENT

```
> document.querySelector("button").style.backgroundColor="yellow";
"yellow"
> document.querySelector("button").style.visibility="hidden";
"hidden"
```

```
document.querySelector("button").style.visibility="visible";
"visible"
```

CSS STYLING:
-KEBAB-CASE
-KEY- VALUE PAIRS
-NO STRING NEEDED



SEPARATION OF CONCERN

Html is for content

Css is to style

Javascript is for behavior

document.querySelector(“”).classList; ===> returns the list of the class that is attached to the element. We can use this to create a new class name, and, add it to class list, and use that class to manipulate the element using javascript

We can add this class in the html using classList.add method with javascript. So when the class added, than the css style will apply

We can also remove classes from the element to manipulate the elements. When I remove invisible from button class, button will be visible again

TASK:

1. Add ‘invisible’ to the button class list

2. Remove ‘invisible’ from the button class list

3. Add invisible to the button class list if it is not listed. Remove if it is already in the class list

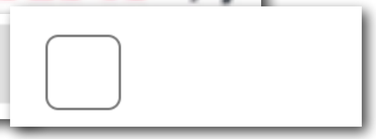
SEPARATION OF CONCERN

CLASSLIST -> RETURNS THE LIST OF THE CLASS THAT IS ATTACHED TO THE ELEMENT
ADD() -> ADDS A NEW CLASS TO THE CLASS LIST
REMOVE() -> REMOVE A CLASS FROM THE LIST
TOGGLE() -> ADD VALUE IF IT IS NOT THERE. REMOVE THE CLASS IF VALUE IS THERE. WORK BASED ON PRECONDITION OF THE ELEMENT

GETTING THE CLICK BUTTON AND ADDING INVISIBLE TO THE TO THE BUTTON CLASS

```
document.querySelector("button").classList.add("invisible");
```

```
<button class="btn invisible">Click me</button>
```



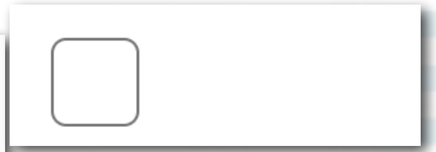
GETTING THE CLICK BUTTON AND REMOVING INVISIBLE TO THE TO THE BUTTON CLASS

```
document.querySelector("button").classList.remove("invisible");
```

```
<button class="btn">Click Me</button>
```

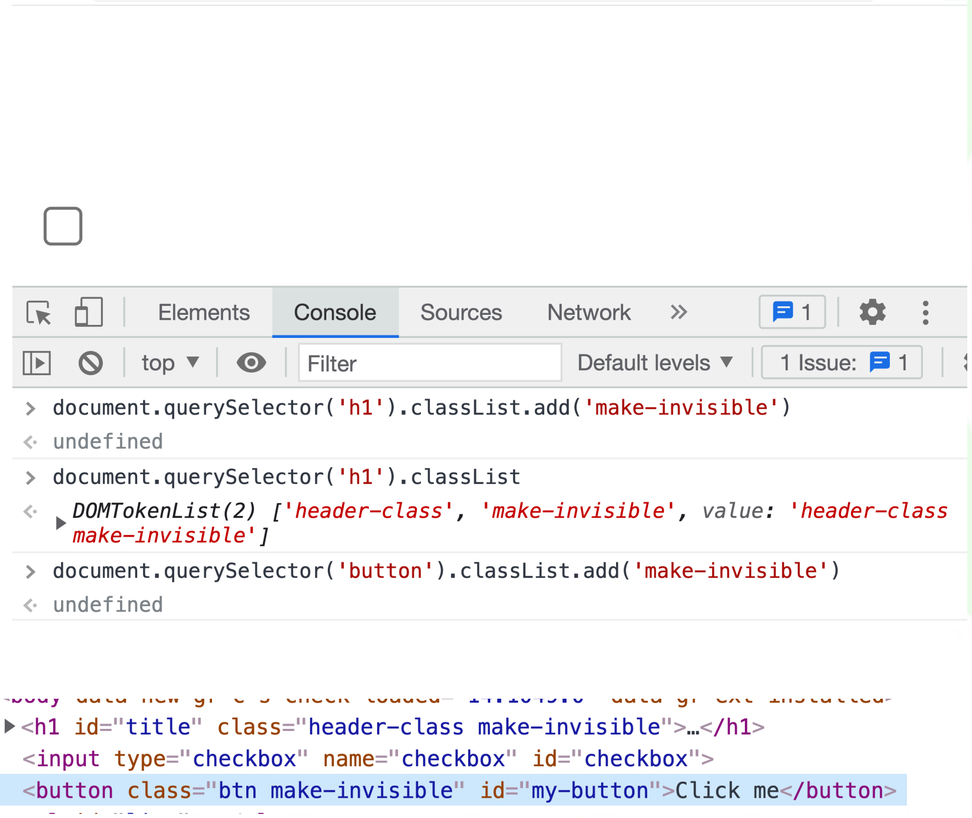
ADDING VALUE IF IT IS NOT THERE. REMOVE THE CLASS IF VALUE IS THERE

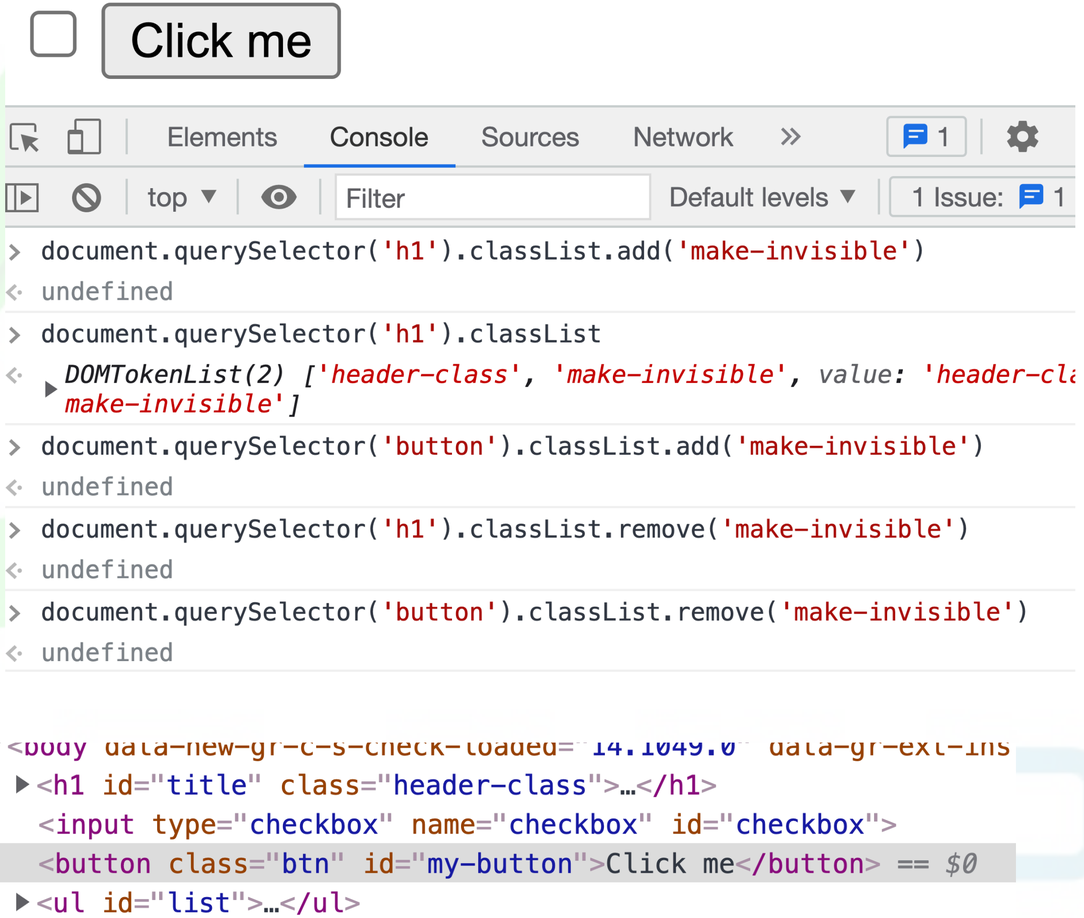
```
document.querySelector("button").classList.toggle("invisible");
true
```



```
document.querySelector("button").classList.toggle("invisible");
false
```

CLASSLIST,ADD,REMOVE CLASS VALUE



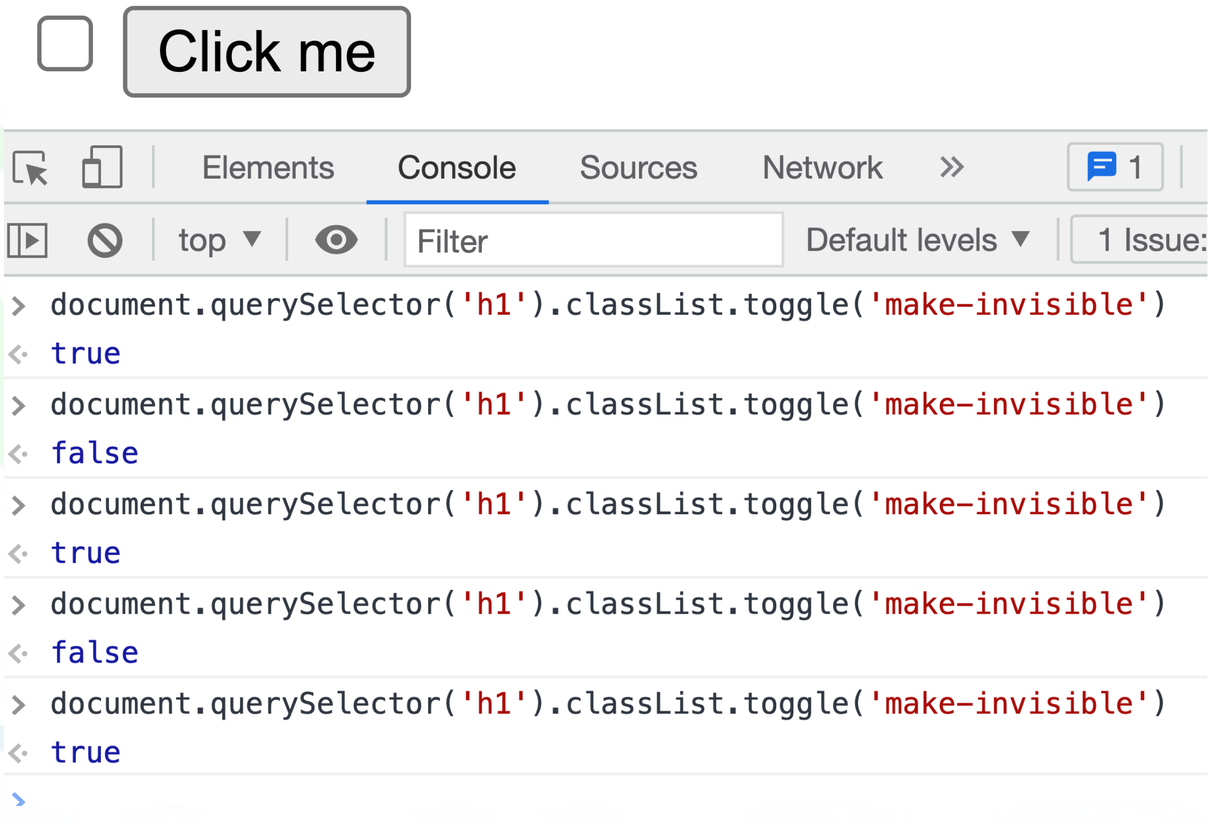


CLASSLIST, TOGGLE

TOGGLE IS USED TO ALTERNATE THE CLASS VALUE

TOGGLE WILL ADD THE CLASS VALUE, IF IT IS NOT THERE

TOGGLE WILL REMOVE THE LASS VALUE IF IT IS THERE



SEPARATION OF CONCERN TASK

1. Create a styles.css sheet and a class named big in styles.css

2. That class will change the element as the following when applied to any element:

Change text size to 8rem

Color should be purple

Text should be bold

Inside Solid border

3. Applied it for the heading in the html using DOM manipulation

classList.add("big") to add the big class to the heading class list, Then the image should be similar

classList.remove(big"), then style of the heading should be normal

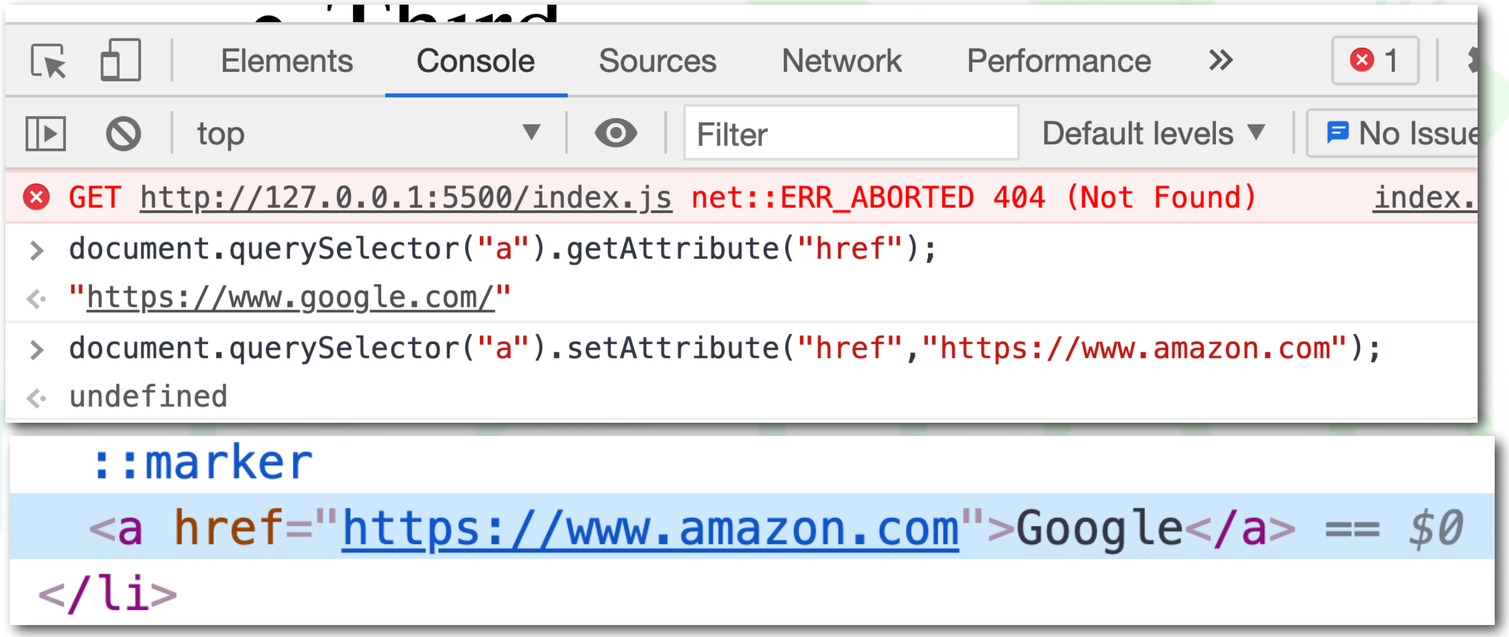
classList.toggle("big"), should alternate the style of the heading

MANIPULATING HTML ELEMENT ATTRIBUTES

getAttribute(“attribute”); ==>gets the value of the attribute
setAttribute(“attribute”,”value”), ==>sets a value tot he attribute

Task:

Change the google link to https://www.amazon.com. When user clicks on google, they should go to amazon



HOMEWORK

Use the dom-project-homework folder

Use the HTML Tree, get the element, then complete the Tasks:

Change the anchor tag(only a tag not li) a color to red

Change “Monthly” to “YEARLY”:

Change 9.99$ to “$100”

Change 99.99$ to “$1000”

Change “2020 - All Rights Reserved” to “None Copyright Material”

Change “No Daily Limit. =>. Email as much as you can

7/24 Access

Yes Tech Support” to “Email as much as you can!”. Rest of the elements should be same

INDEX.HTLM FOR HOMEWORK

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Website</title>
    <link rel="stylesheet" href="styles.css">
  </head>
  <body>
    <h1 id="title">Hello</h1>
    <input type="checkbox" name="" id="">
    <button class="btn" style="color:red;">Click Me</button>
    <ul>
      <li class="item">
        <a href="https://www.google.com/">Google</a>
      </li>
      <li class="item">Second</li>
      <li class="item">Third</li>
    </ul>
        <section id ="pricing" class="pricing">
      <div class="container-fluid r-l-padding">
       <div class="row">
        <div class="col-lg-4 col-md-6 col-sm-12">
         <div class="card text-center">
          <div class="card-header">
           Monthly
          </div>
          <div class="card-body">
           <h5 class="card-title">9.99$</h5>
           <p class="card-text">1000 Daily Mail <br/>
            7/24 access <br/>
            No tech Support</p>
           <a href="#" class="btn btn-primary">BUY</a>
          </div>
         </div>
        </div>
        <div class="col-lg-4 col-md-6 col-sm-12">
         <div class="card text-center">
          <div class="card-header">
           Yearly
          </div>
          <div class="card-body">
           <h5 class="card-title">99.99$</h5>
           <p class="card-text">Daily 10000 Mail <br>
            7/24 Access <br>
            Yes Tech Support</h5></p>
           <a href="#" class="btn btn-primary">BUY</a>
          </div>
         </div>
        </div>
        <div class="col-lg-4 col-md-6 col-sm-12">
         <div class="card text-center">
          <div class="card-header">
           Lifetime
          </div>
          <div class="card-body">
           <h5 class="card-title">199.99$</h5>
           <p class="card-text">No Daily Limit <br>
            7/24 Access <br>
            Yes Tech Support
           </p>
           <a href="#" class="btn btn-primary">BUY</a>
          </div>
         </div>
        </div>
       </div>
      </div>
    </section>
    <!-- Footer starts -->
    <section id="footer">
     <div class="container-fluid bg-blue r-l-padding text-light">
      <a target="_blank" href=""><i class="fab fa-twitter "></i></a>
      <a target="_blank" href=""><i class="fab fa-facebook-f"></i></a>
      <a target="_blank" href=""><i class="fab fa-instagram-square"></i></a>
      <span class="ml-3"> 2020 - All Rights Reserved  </span>
     </div>
    </section>
    <script src="index.js"></script>
  </body>
</html>
```

ROLLING DICE

Create a folder on the desktop and add VSCode: dice-game

Create index.html and paste the code I shared:

Create images folder and add the images I shared

Create styles.css and paste the css I shared

ROLLING DICE

1. Create index.js and link that in the index.js at the end of the body, right before closing body

2. Display the starting images on the webpage: img src should point certain image files:

3. Go to index.js, **generate random number** and use it to **select random dice**. For this, we first need to generate random num, then select random image

4. We should generate a **whole number between 1 and 6**

5. Using this random number to select a random dice image: **dice(randomnum).png**

6. Get the image path using the random dice image: **images/dice(randomnum).png**

7. Select the image using **document.querySelectorAll(“”)[index]**

8. Change the attribute of image1 using **setAttribute(”attributename”,”what you want to change attribute into?”)**

9.TEST YOUR CODE BY REFRESHING THE PAGE. Every time you refresh the page, img1 should change!

10. Generating random img2. Basically do same steps for img2

11. Show the winner. Biggest dice number wins the game

If randomnum1>randomnum2, display “Player 1 Wins”

If randomnum1<randomnum2, display “Player 2 Wins”

If randomnum1=randomnum2, display “Draw!”

Note that we can combine some steps, but we get do this one step by step to make it more understandable

NODE JS

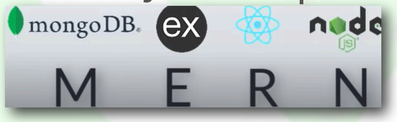
What is Node.js?

It is a server side runtime environment for javascript so you can run js on your machine-not only in the browser

It is a javascript engine that let you run your js on your local computer

Nodejs allows javascript devs to develop backend or mobile with JS. We can run JS on the server-not only client side

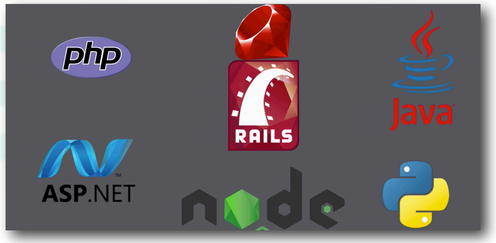
So it is used widely by js developers. You can basically develop full stack application using only javascript with the help of nodes. Some front end devs are called



Mango: database, Ex: expressjs(framework to build application with js, no,- runtime environment)

Npm - node package manager. In java you use external jar files as dependency(database connectors, video convertors, etc). With node js npm package have all the dependencies built in.

Ways to develop backend



NODE.JS

NODE.JS REPL

Open command line

node + Enter: To use get in the node and REPL(read, evaluate, print, loop)

Now you can run javascript like console.log('Hi');, 2+6;, etc

.exit OR control + c: Exit to folder level

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
techproed@techproeds-iMac dom-manipulation % node --version
v14.17.5
techproed@techproeds-iMac dom-manipulation % npm -v
6.14.14
techproed@techproeds-iMac dom-manipulation % node
Welcome to Node.js v14.17.5.
Type ".help" for more information.
> console.log("I am Robot")
I am Robot
undefined
> 10*10
100
> "java"+"script"
'javascript'
> .exit
techproed@techproeds-iMac dom-manipulation %
```

HOW TO RUN FILES USING NODE.JS

1. Create a new folder on the desktop : node-demo

2. Open node-demo folder with VS Code

3. Create index.js file:

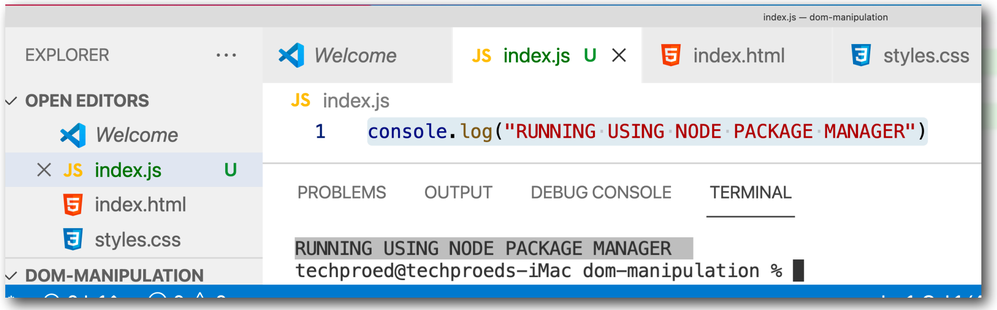
4. In index.js, Add your code, like console.log('Hello World!');

Normally it run on the browser and it is attached with the index.html

5. WE RUN THE CODE USING THIS COMMEND:

How to run a file using node outside of the browser on terminal: **node+ filepath + ENTER**

node index.js +ENTER . This runs index.js in the terminal



NODE MODULES

We can use external modules using node and use them in our projects

Lets use file system and a file into a new file : https://nodejs.org/dist/latest-v14.x/docs/api/fs.html

To use a module, we have to **require** the module

Task:

Create a new text file manually: file1.txt

Copy file1 to a new file to file2 using node by running index.js

---

If file2 is already exist, file2 will be overridden with the copy,

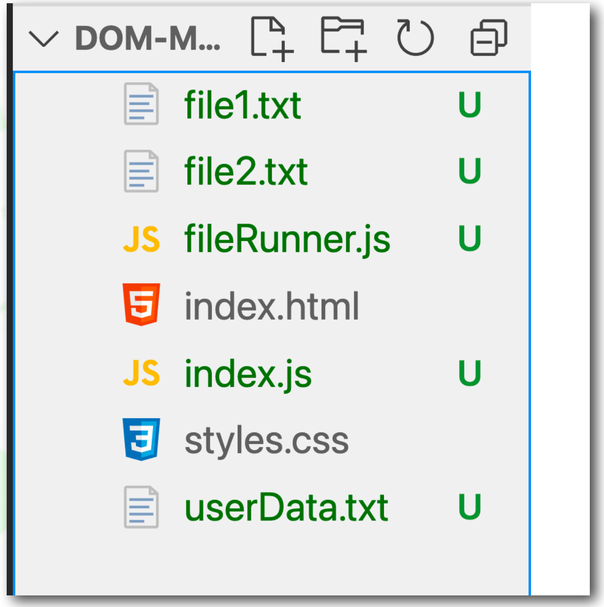
If file 2 doesn't exist, then it will be created

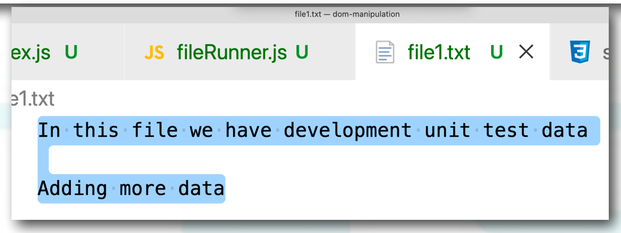
NODEJS FILE SYSTEM SOLUTION

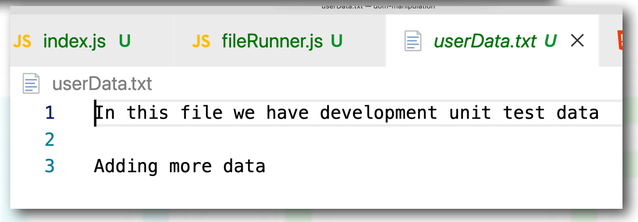
index.js U | fileRunner.js U | userData.txt U | index.html | file1.txt U

fileRunner.js > ...

```
1   // https://nodejs.org/dist/latest-v14.x/docs/api/fs.html#fs_fs_copyfile_src_d
2   //We have to require a module to use in the framework
3   //This is like importing a module in this class
4   const fs = require('fs');
5
6   //I want to make a copy of file1.txt
7   // want to name it as file2.txt
8   //copyFile create a new destination file if the file name doesn't exist
9   //copyFile will override is the name is already exist
0
1   // destination.txt will be created or overwritten by default.
2   // fs.copyFileSync('source.txt', 'destination.txt');
3
4   fs.copyFileSync('file1.txt', 'file2.txt');
5
6   fs.copyFileSync('file1.txt', 'userData.txt');
7
```







NPM

NPM - NODE PACKAGE MANAGER

Npm is external node package manager

Npm codes are based on reusable codes written by other people

Using Npm we can use those reusable codes that is written by other people

Npm has millions of packages that you can install with npm and use(express, react , etc)

You can also create a package and push git so other people can use

To use a certain libraries or package, we must install that package first

Npm automatically installed when we install node

Npm -> Node package manager. It has reusable libraries. We can install and use those libraries.

We need to install npm so that we can install npm package. We already installed npm during node installation

Node -> Node is javascript runner. It lets js run outside of the browser. We need node to develop js application in our IDE.

NPM - NODE PACKAGE MANAGER

LETS USE THE SAME node-demo folder. Make sure you are in the node-demo folder

npm init: to initialize npm

Package name: Enter

Version : Enter

Description: intro to npm

Entry point : Enter

Test Commend : Enter

Git repository: Enter

Keywords: Enter

Author: Enter

Licence : Enter

Is this OK? (yes): Enter

Npm create a package called **package.json** : This is the information we enter when we create the package.

When we use external libraries, and use npm, then npm will create this **package.json file that has the project information.**

NPM - NODE PACKAGE MANAGER

We can import third party npm modules and use them in our application.

Let's demo one.MAKE SURE YOU ARE IN THE FOLDER

Go to https://www.npmjs.com/ and find a package

Website has how to install the module, and how to use the module.

RUN: **npm install packagename**

This commend will install this module and add the **module as dependency in the package.json**

Install superheroes:

```
npm install superheroes
const superheroes = require('superheroes');
superheroes.random();
```

WHEN YOU RUN NPM PACKAGE WHAT HAPPENS

Package.json file will have dependency that has the modules

```
  "dependencies": {
    "superheroes": "^3.0.0"
  }
```

Package-lock.json file is being created. It has detailed module information.

Package-lock.json files shows that we have at least one npm package installed

node_modules has all required module codes in detail

Note: We do not touch this files. We use npm comments to add new modules

INSTALL SUPERVILLAIN PACKAGE

Package.json file will have dependency that has the modules

```
  "dependencies": {
    "superheroes": "^3.0.0",
    "supervillains": "^3.0.0"
  }
```

Package-lock.json file is will include supervillain library info

node_modules will have supervillain module info

Note: We do not touch this files. We use npm comments to add new modules

NPM - NODE PACKAGE MANAGER

Try to install supervillains and use it

to write a mini code

```
// //WE must require a module to use in the applicaiton
// const fs = require('fs');
// // When there is no destionation file,
// // Node will create one new file
// // If destination exist then it will override(replace)
// fs.copyFileSync('file1.txt','file2.txt');
const superheroes = require('superheroes');
const supervillains = require('supervillains');

const mySuperHero=superheroes.random();
console.log(mySuperHero);
const mySuperV=supervillains.random();
console.log(mySuperV);
```