Basics for website → HTML, CSS, Javascript
(must) (Beauty) (Brain)

⊙ How website work?

client request for website ←

web server ⟷ client [Browser]

→ server returns Response (HTML CSS JS)

it has own IP address

↳ Backend of website placed on this IP addr

[Backend]
eg. Python
NodeJS
PHP

⇒ Backend Compute and gives result (HTML CSS JS)

CSS JS

Envelop = HTML

This will be given as Response to Request for website

⇓

Web Browser will Run the document html and it will show you results (Browser manipulate file into visual data)

⊙ HTML ⇒ HyperTexet Markup Language

↳ it is basically a standard markup lang. For giving static skeleton to web application & websites

⊙ CSS ⇒ Cascading Style Sheets

↳ Style sheet language that used to handle presentation of web page containing HTML

↳ Modification, Beautification (eg. Animation)

Let's understand HTML Boiler Plate

1. <!DOCTYPE html> ⟶ // Type of Document HTML

2. <html lang="en"> } // opening & closing of HTML
   </html>
   ↳ // lang is attribute that means Language = "English"

   ⊙ Type of documents Browser understand
   ↳ html, xhtml, etc

3. <head>
   <meta charset="UTF-8"> // head contains various meta tags
   <meta ——— >
   </head>
   <body>
   —— Write Body content to Display"
   </body>

DATE 11 may

⊙ Along with Browser installation you get small parser (eg. HTML Parser)

⊙ hi
↳ content
↳ Bold Tag
This syntax is understood & interpreted by HTML parser

• 1) Deal with HTTP
2) understand HTML
} needed for website

⊙ Job of web server:
Generate something that your browser will understand

⊙ Html is understandable by HTML parser to the Browser

⊙ code is written with .html / .htm extension
who understands this file & syntax ⇒ Ans: Program called HTML parser (This program exists in Browser)

(you can write on server some prog. like JAVA / .NET / PHP to generate HTML files contents)

1. <html> <!-- start of file -->
<head>
<title> Sunbeam </title>
</head>
<body>

Head tag used for - Metadata, scripting, styling etc.

⊙ Favicon icon eg. ⓖ
.ico Files (image with extension ico)

⊙ Newet line : ~~
~~ ~~</br>~~ Tag
 OR
Some tags are self closing tags
They don't have end tag

For some Browser's HTML parser .... it is ok
if we dont self close and keep the br tag
is like open .... it would still work.

⊙ Header : <h1> Heading </h1>
<h2> Heading </h2>
<h3> Heading </h3>
} Difference in size of heading

h1 → Big Bold
h2 → smaller than h1 & respectively

eg. h1 to h6 are come with Auto Break
such tags are called Block Tags

⊙   : one numeric space
↳ This is keyword.

eg.   ;   ; Hi <!-- 2 space -->

Day 2 Classwork

DATE 12 May

Requests

http://---/demo.html ⟶ live server → checks what is requested?

Browser (client) ⟵ finds demo.html and sends file back to client/ Browser.

HTML parser ↓ reads HTML line by line

- Tag for image : <img src="picture.jpg" alt=" ">
- you have just requested classwork2.html / demo.html but Browser internally read line by line and internally shoots a call for image inside it: (demo.html requested to browser to shoot call for img file)
- If you have moved your images in image folder
  <img src="image/pic1.jpg">
- height & width
  <img src=" " height="100px" width="100px">
  (pixel)
- <hr> : Horizontal Ruler
- Anchor Tag (For Link): Contactus
- New Tab For Link :
- How to make image clickable

```
<a href="home.html">
    <img src=" " ---->
</a>
```

⊙ <P> </P> paragraph Tag

⊙ Tabular Representation of data

① <table> </table> Tag → without any size table
specify width & height

② Table Border : <table border = "1"> </table>
↑ This is shortcut // NOT recommended
without understanding border
in styles

③ <Table border = "1" height = "500px" width = "500 px">
<Tr> → Table row
</Tr>
</Table

④ <Table border = " " height = " " width = " ">
<Tr>
<Td>] Col 1
</Td>
</Tr> <Td>] Col 2
</Td>
</Tr>

Row 1

eg. <Tr> <Td> Name </Td>
<Td> Address </Td>
</Tr>
<Tr> <Td> Saurabh </Td>
<Td> katraj, Pune </Td>
</Tr>

⑤ <table style = "border : 1px solid" > → CSS

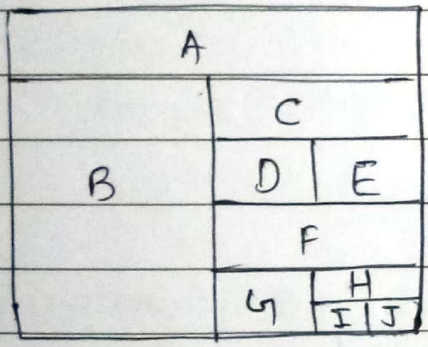
⑥   → To shift Table at center / content at center

⑦ <Th> </Th> → Table Header
<Tr> <Th> </Th> </Tr>

⑧ <td colspan = "2"> </td>
↳ coloumn will occupy size as two coloumn.
<td rowspan = "2"> </td>

⑨ Merger of cells : (IF size of cells are same

⑩



(~~Assigment~~) (Labwork)
using tr, td, colspan, rowspan

⊙   → unordered list // disc // square // circle

eg.   This is unordered list

o/p: • This is unordered list // Bullets

⊙   → ordered list

  ordered list

o/p → 1. ordered list // Numbering

<ol type = "I" >

<ol type = "A" > etc.

⊙ you can write list under list

⊙ Tables

<Table> </Table> → Tag.

1) <thead> </thead> → table head
2) <tbody> </tbody> → Table body
3) <tr> </tr> → row
4) <td> </td> → coloumn
5) <th> </th> → heads in Table

eg.

```
<table>
<thead> <tr> <th> Name </th>
             <th> Id </th>
             <th> Job </th>
        </tr>
<tbody> <tr> <td> Saurabh </td>
             <td> 63 </td>
             <td> Developer </td>
        </tr> </tbody>
</table>
```

DATE 13 May

⊙ Style
- Inline ; Internal ; External

```
<p style = "background-color : pink ; color : blue ;">

<p class = "myparastyle" > </p>
<style>
    .myparastyle
    {
        background-color : yellow ;
        Color : blue ;
    }
</style>
```

Internal Style

⊙ Create a mystyle.css file
define →

```
.myparagraphstyle
{ background-color : yellow ;
    Color : blue ;
}
```

⊙ General Purpose Attributes

① Id

eg. <tagName id = "attribute value" >

identifier ↵ ↳ unique Name of element

it is useful to use in Backend programs

e.g. <P id = "First" > This is First Blog - -

value

In CSS : # First {

Colour : blue;

}

[In CSS using se
& id value we,
apply attribute
~~wherever~~

② Class

eg. <P class = "(value/Name)" > Paragraph started (IF

⊙ आप एक से ज्यादा elements को same class दे सकते

Same class value

eg. <P class = "First" > This First Blog

<P class = "First" > This is second Blog

Blue color to both

In CSS :

. First {

Colour : blue;

}

O/P :

This First Blog

this is second Blog

(Uniform Resource Locator)

⊙ Hyperlink — URL

< a href = " http : // - - - - " >

Anchor

URL → absolute (complete path)
↳ Relative (current path)

⊙ Moving into File locations

1. Moving into Folders

→ Folder name / File_Name . extension "

2. Files in same Folder (if File is in current Folder)

→ write direct name of File with extension

3. Moving out of Folder

→ .. / FileName

⊙ open link in New Tab

< a href = " www.google.com " target = "_blank " > Google(li

⊙ Video File

< video src = " " width = " 400 " >

< video src = " " width = " " autoplay >

< video src = " " width = " " controls >

< video

< video

(control need for pause play)

loop >

Poster = " " >

↳ [it is for thumbnail]

⊙ Internal CSS:
Write style tag inside head Tag

e.g
```
<head>          → selector
<style>  P {
            color : yellow;
         }
</style>
</head>
<body>  <P> Messege will show in yellow </P>
</body>
```

⊙ [Inline-CSS > Internal CSS] Priority order when Both applied on same element

⊙ External CSS :
make .css file → write code

~~[External > Internal CSS / CSS] priority~~

⊙ जो line बाद मे लिखी है internal/externa में से that will be apply

⊙ But if you write → !important in Any CSS that will get highest priority

⊙ CSS element selector ——→ p { border : 2px solid red; }
CSS Id selector
CSS class selector
CSS grouping selector

⊙ To apply particular style to particular element

```
#redele { color : red ;
        }

<P id = "redele" >                    </P>
```

eg.
```
<head>
    #redele { color : red ;
              background - color : blue ;
            }
</head>
<body>
    <p class = " redele" id = "first">          </P>
    <p class = "Blele" >    </P>
    <p class = "redele">    </P>
```

• For class ←

⊙ # ⟶ For Id
. ⟶ For class

⊙ Grouping selector :

```
eg.   Footer, span {
          color : pink ;
          background - color : Black ;
      }

<span>          </span>
<Footer>        </Footer>
```

→ High level language, prototype based object orientation

Brendan eich → Father of Javascript

Java	Javascript*
Compile Time Lang.	Interpreted Lang. (Line by line code executable)
	object based lang - scripting lang
	Polymorphism, Encapsulation, Inheritance } Features
	→ ECMA International standardised
	→ both Frontend & Backend
	→ Frameworks Angular JS,

⊙ <Script> </Script> → For inline Javascript
<Script src="File.js"> </Script>

⊙ In .JS File

```
let n1 = document.querySelector('button');
n1.addeventListener('click', showMsg);

Function showMsg()
{  alert("Namaste World");
}
```

⊙

```
('click', inputMsg);
Function inputMsg() {
        let n1 = prompt("enter Name");
        named.textContent = 'Roll No. 1' + ~~Name~~ n1;
}
```

↳ client side scripting lang.
↳ make web pages alive
↳ programatically perform actions within page
↳ Initially called Livescript

JS also helps to send data to server
ajax request → Asynchronous Javascript & XML
↳ it helps : बिना page को reload किए हुए, page के content को dynamically change करते है.

As a object available to JS → DOM → Document object Module
↳ it may contain - HTML, style, body, div, nav etc

⊙ Use of JS
1) JS can execute not only in browser, but also on server
2) client side & Server side lang.
(Node JS → for Backend)

⊙ JS - in Browser side
1) Safe low level CPU permissions
2) Add new HTML & change existing HTML from DOM
3) React to events (actions) (eg. response to request)
→ response from server
→ Mouse Move
→ Page populate, key press etc
4) AJAX Request
5) Get & set Cookies

⊙ Can't Do in Browser
1) Read / write to & from computer Hard disk

⊙ var → global scope by defaut
if we declare var in function then restricted

⊙ let → Block level scope
eg.
```
let a = "u";
{

}
```

⊙ eg.
```
<script> var string1 = "Hi";
var string1 = "Hello"; // o/p: Hello
Console.log (string1);
```
redeclared

```
let a = "u";
{
    let a = "z";
    console.log (a); // z
}
Console.log (a); // u
```
global

Block Scope

⊙ Const a = "This cannot be changed";
a = "changed"; // Error

⊙ let age = 34; (If Else)
```
if (age > 18)
{
  Console.log ("you can drink water");
}
else {
    Console.log ("Drink Milk");
}
```

```
if {
   }
else if (   ) {
              }
 else {
   }
```

```
let age = 2;
if (age>18) { console.log(" ");
            }
else if (age==2) { console.log(" ")
                 }
     else { console.log(" ");
          }
```

⊙ switch case

```
const  c = 45;
Switch (c) {
          case value:
                    break;
          }
```

~~const~~

```
const  c = 45
switch (c)
{
case 41 : console.log();
          break;
case 45 : console.log();
          break;
default :
}
```

<script>

var A = 34; or let A = 34;
let B = "String";
let C = true;
let D = undefined;
} Primitive

① let employee = { name : "Saurabh" , (key, value)
salary : 10000 , (key, value)
Job : Developer (key, value)
}
} object

console.log(employee);

employee.name
employee['name']
↳ o/p : Saurabh

② let names = [1, 2, 4, "Harry", undefined];
console.log(names);
console.log(names.length);
} Array

names = names.sort(); → to sort array

names.push("Added in Array");
console.log(names);
↳ o/p : [1, 2, 4, Harry, undefined, Added in Array]

③ let A = new Array(1, 2, 3, 4, 5);

④ let B = new Array(25);
↳ now this will become size
∴ console.log(B.length); → 25

Day6 Lecture

DATE 18 May

- let A = "100";
  let B = parseInt(A); // B = 100

- Functional Programming in JS
- Object Oriented Prog. in JS

```
Function check (success, Failure)
{
  if (10>2)
    { success();
    }
    else {
        Failure();
    }
}

  Function F1() {
        console.log (" 10 is greater than 2");
        }
  Function F2() {
        console.log (" No, its Miracle);
    }
  check (F1, F2);
```

Call Back Func^n's
Check Func^n is calling function

- you can have Nameless Function called Anonymous function

```
eg.   () {
        console.log (" 10 is greater than 2");
      }
```

- Arrow Func^n similar to lambda func^n in Java

```
check ((Parameter) => { console.log ("      ");};
                  () => {console.log (" ");});
```

① Function Add(x,y) // Defaut way
{
console.log(x+y);
}

② var Add = Function(x,y) // Anony. Funch holding in variable Add
{
console.log(x+y);
} variable Holding Funct
Add(10,20);

③ var Add = (x,y) => // Variable Holding Arrow Function body
{
Console.log(x,y);
};
Add(10,20);

④ A Function can Return Function

```
Function Execute(choice)
{
    if(choice==1)
    { return function(x,y)
        { console.log(x+y);
        }
    } else
    {
        return function(x,y)
        {
            Console.log(x-y);
        }
    }
}
```

```
function DoThis (choice)
{   if (choice == 1)
    {
    return (x, y) => {
                        console.log (x + y);
                        }
    } else {
            return (x, y) => {
                                console.log (x - y);
                              }
           }
}

var ThisFunc = DoThis (1)
    ThisFunc ( 10, 20) ;
```

⊙ Functional programming in JS : Func^n concept centric

1. we can write Normal Func^n
2. we can pass Func^n as parameter - which will be called as callback Func^n
3. We can hold a Func^n body in a variable & can call that variable like Func^n while calling
4. we can return Func^n from Func^n (choice)
5. Func^n can be Anonymous
6. Func^n can be Arrow Func^n
   (It Feature From ECMA std version 6 i.e 2015 onwards)

⊙ OOP using func keyword - Constructor Func syntax ECMA - Parameterized Constructor

```
var person = function (no, name, address)
{
    this.NO = no;
    this.Name = name;
    this.Address = address;

    this.print = function()
    {
        console.log(this.No + this.Name
                    + this.Address);
    }
}

var P1 = new person (1, "Sawabh",
                     "pune");
var P2 = new person (2, "Preetam",
                     "Pune");

console.log ("Name is" + P1.Name);
P1.print();

P2.Address = "~~chennai~~ Mumbai";
P2.print();
```

// we are writing class Here - person
// using - constructor function syntax

// in this func^n we are passing no, name, Addr.
// these are parameters to constructor
// we are holding these parameter values in getters & setters called as NO, Name, Address

⊙ Inheritance

```
var employee = function (no, name, address, Dept)
{
    //this.No = no;
    // Name = name;
    // Address = address;
    this.Dept = dept;
    this.print = function()
    {
        console.log("this.Dept");
    }
}
```

common with person

(Dot Underscore Underscore)

// Employee is a Person ; Reusability

Employee.__Proto__ = Person; // This means Employee inherits Person

debugger;

var e1 = new Employee (1, "Saurabh", "Pune", "Developer")
e1.print();

① Before ECMA - 5

```
var person = {
        No = 1;
        Name = "Saurabh";
        Address = "Pune";
    print = function () {
        console.log (        )
        }
    }
var P1 = person;
var P2 = person;
P2.Name = "preetam"
```

Same person object

[we can not do new]

② class syntax of class creation in ECMA 6 (2015)

```
class person
{   constructor (no, name, address)
    {
      this.No = no;
      this.Name = name;
      this.Address = address;
    }
    print () { console.log ( this.No ——— ) ;
    }
}
```

```
var p1 = new person (1, "saurabh", "pune");

class employee extends person
{
    constructor( no, name, address), dept)
    {
        super(no, name, address, dept); this.dept=dept
    }  print() { console.log( ____ this.dept);
}
var e1 = new Employee (1, "saurabh", "Pune", IT");
    e1. print();
```

Class Day 6

DATE 18 May

- ECMA Script 6 From 2015 ES6
  Important changes [let, const, new array function]
- more funcn from document
- JSON basics
- Using JSON with XMLHttpRequest object
- jquery.js basics
- css 3/ responsive css - bootstrap

- <script> // ECMA 6 changes
  // let & const keywords
  // understand the need for let & const

```
var counter = 0;
function change() {
    counter = counter + 1;
    console.log(counter); // 1
}
change();
console.log(counter); // 1
```

- Always preference to local variable in function

```
var counter = 0;          // global
console.log(counter); // 0
function change()
{
    var counter = 1;      // local
    Console.log(counter); // 1

}
change();
Console.log(counter); // 0
```

Change in ECMA 6

```
var c=0;
Con.log(c);  // 0
Function change()
{
    coun c=1;
    C log(c); // 1
}
change();
Con log(c); // 1
```

Here variable is Not declared

Recommendation: Use 'let' to declare local scope variable

lambada / Arrow Func^n is part of ECMA 6

```
Var add = (x,y) => { console.log (x+y) }
add(10,20);
```

map function : Use For Iteration

```
Var student = ["A", "B", "C", "D"];
student.map(DisplayStudents);
```

* Internally *

```
Function Map (student)
{
For (let i=0; i<student.length; i++)
    {
    let student = student(i);
    DisplayStudents (student);
}
Function DisplayStudents (student)
{
    Console.log ("Hello" + student);
}
```

⊙ student.map((student) => {
another way ↑ console.log("Hello" + student);
});

⊙ Changes
1) let & ~~var~~ const
2) arrow func^n syntax
3) new array functions : Map, Filter
4) Spread operator i.e triple dot
5) Template literal

⊙ Spread operator

⊙ var A = ["Sau", "Pree", "shiv", "Rani"];
var B = ["Monday", "Tuesday", "Thur"];
var C = [24, 30, 20, 42];

var All = [...A, "Raj", ...B, "sat", ...C];
(dot dot dot)

⊙ var person1 = {no: 1, name: "saurabh", add: "Pune"}

var person2 = {...person1}

⊙ Filter function
↓ it will also iterate through array & work upon condition

var students = ["Ra", "sa", "Pree", "Moh"];
students.filter(student) => {
~~console.log~~
return (students.includes("r"))
});

Class Day 7

DOM Manipulation

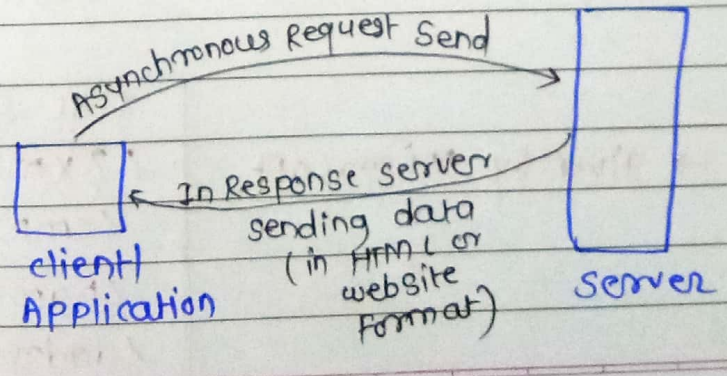
eg.
```
var cnt = 0;
function call()
  { cnt = cnt + 1;
    console.log(cnt);
  }
function callMe()
  {
    //call();
  }
window.setTimeout(call, 5000);
// execute call funcn after 5 sec
```

execute call fun only once

- setInterval : calls the callback funcn continously after a given interval
- setTimeout : calls callback funcn only after given interval
  we can write logic to call setTimeout & make it behave like setInterval

- XML → eXtensible Markup Language
- JSON → Javascript Object Notation
- JSON → Format to transfer data from client to server & from server to client
  - Dynamic page → on your request you will get response

eg.



Developers used JS to send data & get data in String Format

- Techno used on server side → Java, ASP.Net, PHP
- request ⟷ Response (In text/plain format)
- Emp object
  - empid
  - ename
  - Sal
  - dept

  → server will send data in string Format (object Format)

  How to fetch data when single value / single element?

  JSON Format ← you can send this data in the object format itself in Javascript for that we will use Javascript object Notation (JSON)

```
var Emp = {
    empid : "1",
    ename : "Sourabh",
    Sal : "600000",
    dept : "Education",
}
```

- GSON
- JACKSON

} Libraries to convert Java to JSON & JSON to Java

- XML → given by Microsoft

XML Syntax

```
<?xml ?>
<emp>
<name>mahesh</name>
<address> pune </address>
</emp>
```

JSON → storing & transporting data
↳ Data to send from server to webpage
↳ you can extract data in format of Text in JSON
↳ If you want plain Text
↳ We convert all the objects in JS & finall converted into text
↳ open Data Interchange

Advantages: 1) Lightweight compare to other Data Interchange options
2) Less verbose : More compact style than XML & often More Readable
3) XML parser takes lot more time because DOM manipulation libraries that required more memory to handle large XML files
4) JSON less Memory → Fast → High parsing speed

① JSON uses a Map Data Structure rather than XML Tree structure
Map DS → key & value pair (little restricted) but you can get predictable & easy data model

⊙ [Both of these used for the same purpose for open Data Interchange & these are used to convert data into texts when Communicⁿ betⁿ client & server takes places both of these are top choices.

⊙ XML is since long then why we move to JSON?
→ adv. of JSON

JSON vs XML

↳ JSON is parsed into ready to use Javascript object which makes it way easier & simpler.

* ↳ Just fetch a JSON string and then do the JSON.parse for JSON string and that's all you got to do and that's exactly why people started preferring JSON over XML because it makes your work easier

↳ Difficult to parse than JSON

* ↳ First fetch XML document then use XML DOM to loop through the document & then extract values & store in variables

⊙ JSON syntax Rule

JSON syntax is basically a subset of JS syntax

1) How to define object in JS?

{ name: "John" } → in Javascript
{"name": "John"} → in JSON

⊙ data should always be in key & value pair
⊙ one you add more data, seperated by 'commas'

e.g. {"name": "John", "age": 23}

XMLHttpRequest (Helper object)

→ How do I ask server to have dynamic data, we will shoot call from server to XTMLHttprequest

→ Javascript can use XTMLHttprequest & can create a packet & can send the packet to server & expect a reply

→ Earlier we use to send data to server as a 'string' using delimiliters → then later on XML came in picture → Now we are using JSON

→ We will ask Javascript let us use this object from browser → let us use it to create a packet → send this packet to website → let's get the reply packet → unwrap the packet → find the data → present the data on screen

(To avoid Hardcoding & Generate Dynamic Data on page)

eg. Facebook, Insta, etc.

```
<script>
```

variable name

① var helperFromBrowserToCallServerForJS = new XMLHttpRequest();

// XMLHttprequest is given by browser just like document, window, etc.

② helperFromBrowserToCallServerForJS.Open("GET", "https://m...");

// Browser is Not taking decision to get packet, I am asking Javascript to ask browser to get packet through code

③ helper fords.send(); → packet is sent

(Helper object)

⊙ XMLHttpRequest

→ How do I ask server to have dynamic data, we will shoot call from server to XMLHttpRequest

→ Javascript can use XMLHttpRequest & can create a packet & can send the packet to server & expect a response

→ Earlier we use to send data to server as a 'string' using delimiliters ⟶ then later on XML came in picture → Now we are using JSON

→ We will ask Javascript let us use this object from browser → let us use it to create a packet → send this packet to website → let's get the reply packet → unwrap the packet → find the data → present the data on screen

(To avoid Hardcoding & Generate Dynamic Data on pages)
eg. Facebook, Insta, etc.

```
<script>
① var helperFromBrowserToCallServerForJS
        = new XMLHttpRequest();
// XMLHttpRequest is given by browser just like document, window, etc.

② helperFromBrowserToCallServerForJS.Open("GET", "https://www...");

// Browser is Not taking decision to get packet, I am asking Javascript to ask browser to get packet through code

③ helper ____ forJS.send(); ____ packet is sent
```

variable name

Bootcamp

REACT → Library

⊙ JS for user experience such as action, events, validation etc
↳ To make interactive UI (DOM)

⊙ JS is wrapped in Browser

⊙ Why Need React?
↳ we can do using JS also, then why react
↳ For good developer experience
↳ Virtual DOM
↳ component Reusability

⊙ What is DOM?

⊙ CDN Links

⊙ Ready state Property
↳ getter - setter
↳ From this object we will tell current stage we are at in terms of request & response

→ // since it is yet to start creating request packet .... readyState = 0

(1.1) helperFromBrowser ____ JS.onreadystatechange = () => {
// this code here will be called whenever readystate changes
console.log(helperfrom ____ JS.readyState);
}

Here HTTP packet is getting created which will send a GET request to this website "https://www ____ "

→ // Packet is created Now ∴ readyState = 1

→ // since packet is sent ... readyState value is 2

→ // Now the packet is sent we are waiting for a reply ∴ readyState =

⊙ Jquery → Javascript Library

* Basic DOM :

Document
<html> <head> <title>
</title>
<body>
</body>
</html>

* JS is scripting language to interact with DOM
* Jquery has ability to access each node & modify it

* each browser has slightly diff. DOM interface
(chrome vs explorer vs Mozila vs opera vs safari)

* Jquery is like a Manager who knows each of workers (i.e Browsers here) - knows behaviour of each & every browser.

* Jquery Remove dependency to write code for each & every individual browser

⊙ Jquery → JS library that simplifies how to traverse HTML documents, handle events, perform animations & AJAX for web Development

↳ Free & Open Source JS library (2006)
↳ Cross-Browser support
↳ light weight
↳ Improves developer efficiency

Why Jquery ?

1) jQuery is a JS library designed to simplify client Side Scripting of HTML
2) create Dynamic web pages (Dom applicn)
3) Faster loading pages
4) animated pages like Flash

HTML Document Types

HTML 4/XHTML 1.0 – DOCTYPE
HTML5 - Doctype

Css selectors → select element based on names, Attributes, position

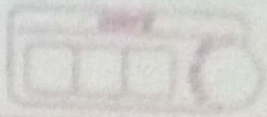
1) TagName selectors
2) ID Selectors
3) class Selectors
4) Pseudo-classes
5) Attribute selector
6) Multiple selector
7) Selector based on Relationship

Jquery Object ($)

→ jQuery (<code>) : It is a jquery funcn
→ $ (<code>) : Alternative jquery funcn
→ $(document) : jquery funcn usually takes a single argument in this example "document"

→ jquery object is like an Array, it also contains object methods like :
  → Length
  → Context
  → selector

Class Day 7 - Evening



Browser understands HTML, CSS, JS, json, images

→ when you call anything from Browser & ask anything to server

Suppose you ask
- .jsp
- .php
- .asp (C#)

↓ convert to

```
<html>
<div> - - -
</html>
```

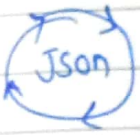
Server
→ Find what is asked, see if browser is able to understand i.e. if it is html/css/JS if Not ↓ server can process file & generate final o/p which is in HTML, CSS, - - -

```
<div>
<div>
  + js code
  + XHR HTTP request
```

Short call to server

return JSON code

JSON will iterate in Browser & generate UI or div within div



Industry Requirement

→ eg. write a code that fires delete, update query on database using Java or .Net

→ get the data from database

→ convert that data into json Format & wait till jquery or javascript or angular or React gives call to you

→ And then give that JSON to next person

Web Utilise / Web APIs

⊙ Jquery Studies in this class
- ↳ syntax
- ↳ methods
- ↳ Functions
- ↳ How to use it as alternative to dynamic css
- ↳ Events

(use cheatsheets)

jquery is eventually Javascript

class Day 8 - Morning

javascript → validation.js → .html File
javascript → jquery.js → .html File
} Manipulate HTML elements

- jquery.com → API documentation → All functions (syntax) (sample codes)
- regres.in/api/users → complete json code
  we have learnt to ghost call via javascript to get the data using xmlhttprequest object
- How To xmlHttpRequest indirectly using jquery

```
<input type="button" value="Get data server" id="btn">
<script src="jquery-3.6.0.min.js"></script>
<script>
    jquery('#btn').click(function() {
// If we have to use Javascript for XMLHttpRequest call
① var helper = new XMLHttpRequest();
② helper.open("GET", "https://regres.in/api/users")

③ helper.send();

helper.onreadystatechange = function() {
        if (helper.readystate == 4 &&
            helper.status == 200)
{
var userData = helper.responseText
All_users    JSON.parse(helper.responseText);

All_users.data.map(user => {
    alert("Hello" + user.first_name);

});
}
```

AJAX call →

check if we got a reply

AJAX → Asynchronous Javascript & XML

// Asynchronous = JS engine is not waiting for a reply from server for the request packet sent using .send() method below

// XML: Data expected used to be in XML in string Format .... bu today we Normally exepect JSON

⊙ If we have to use jQuery For XMLHttpRequest (or) AJAX call :

case sensitive parameters

```
$.ajax({ // internally using XMLHttpRequest
    url : "https://reqres.in/api/users",
    type : "GET"
    contentType : "application/json",
    success : function() { }
    error : function() { }
});
```

What to call → url

type of call → type

open() & send() replaced here in Ajax with this syntax

This Interna use json.pa

```
success : function (allUsers) {
        debugger;
allUsers.data.map((user) => { alert("—"); });
});
error : function (errDetails) {
        debugger;
}
```

we can get reply successfully or we can get error Perform functions accordingly

IF success → perform

if (readyState == 4 && status == 200)

7) jqueryui.com → Interactions
→ widgets → view source
→ Effects [copy code & paste in our .html file]
→ Utilities

8) Bootstrap Started
getbootstrap.com → examples

```
<div class="container">
    <div class="jumbotron">
```