**JS-1**

Netscape develops JavaScript in 1995 (Nearly in 10days)

JavaScript is light weight scripting language, client-site scripting language or scripting language. It is used to **implement functionality and behavior** in our webpage.

What we can do with JS? -> We can build web app, mobile app, network app, CLI tools and games.

Script is set of instruction in which we have to tell what to do.

**JS engine** is an environment in which we can run JS and it is not a compiler.

In firefox there is SpiderMonkey JS engine and in chrome v8.

Today JavaScript is both **client (browser) and Server-Side (outside browser) language**. (C++ integrate with JavaScript to make it Server-Side language which is called **node**.)

Script tag in which we can contain all the JS code and it is used for client-side scripting.

Console window is a tool from which we can see the flow of a code. Basically, it handled network request.

First we render all the html code and then put **the script tag at the end**. It will also improve user experience. (Best practice)

Variable is named memory location.

**Var and let is used for variable declaration the difference is scope of that variable. Var is used for global scope and let is used in only in that code blocks or we can say let is block scope variable.**

**Re-declaration of variable is not possible in let keyword. In case of var re-declaration of variable is possible, it will consider the last declaration.**

JavaScript is dynamic typed language. (We can store any type of data using let or var keywords.)

Constant value does not change during program execution. Constant variable cannot be reassigned.

Variable naming rules: -

* Cannot be a reserved keyword.
* Meaningful variable name.
* Cannot start with a number.
* Cannot contain space and hypen(-).
* Use camelCase. (ex- myfirstName).

let a, b, c; // creating multiple variable.

**Primitive Types** – String, Number, Boolean, undefined and null.

**Undefined** primitive data type means we did not define that variable, we only declare it. And **null** means that we defined that variable to null.

**Reference Types** -> Objects, Arrays and functions.

Multiple variable or property agar link hai ek dusre ke sath to hum use object bana sakte hai. (Top level Entity) Object have some property and behaviour.

Object property can be accessed by using **dot notation and bracket notation**.

**Here in array we can store a list of items whether it is number string Boolean**. (Due to dynamic typed language.)

**In strict equality(===) value and data type both should be same but in loose equality(==) only the value should be same.**

**Operators:**

* Arithmetic
* Assignment
* Comparison
* Bitwise
* Logical

Ternary Operator -> condition? value1: value2;

**Logical Operator->**

* And && -> all condition has to be true.
* Or || -> any single condition is true.
* NOT ! -> toggle the value.

Logical operator working with non-Booleans. (1:35).

Bitwise operator -> & and |. (1:43)

Control statement -> if else and switch. (1:47)

Loops: if we want to do repetition of task, we can use loops.

* For loop
* While loop
* Do-while loop
* What is infinite loop
* For-in loop
* For-of loop

**JS-2**

Object - A real world entity, collection of key-value pair, instance of class, same property lo wrap kr diya (Multiple linked variable), like structure.

An entity which has some property and behavior. (example- square has side as a property and area as function)

Function ek tarah ka object he hota hai…

**Function is a program and sub-program which can perform well defined task.**

**If you are creating any function inside a object, then the right term to say is method not function.**

let a ={ } -> means ‘a’ is object, it showing an empty object. { } this is object literal.

**Object Creation – HOW? (Bulky code and so much bugginess)**

1. Factory Function - In factory function we create function in which it will create an object and return it.
2. Constructor function - constructor function is nothing but only initialising or defining property and method of an object. Here we don’t return any value like factory function.

**Dynamic nature of object** -> means in object we can add any property or delete any property.

**Constructor Property ->** Function is also an object and every object has a constructor. (1:05)

**Types in JS:**

1. **Primitive or value types - > copy bante hai on eqauting**
2. **Reference types or object -> same address par point krte hai on eqauting**

**Primitive are copied by their value and References are copied by their address.**

**Iterating through object:**

1. for-in loop -> using this we can iterate key value pair of an object
2. for-of loop -> we can apply only on iterables.

**How to find properties if exist in an object or not?**

if('color' in rectangle){

    console.log('present');

}

else{

    console.log('absent');

}

**# object Cloning through iteration, assign and spread.**

// we can copy multiple object and source into destination using object.assign()

SEE CODE JS CLASS 2

Garbage Collector-> it finds those variable and constant which were not is use and deallocates it automatically. We have no control over Garbage collector. It runs in background.

**JS-3 (In-built object and arrays)**

**Math() -> in built object**

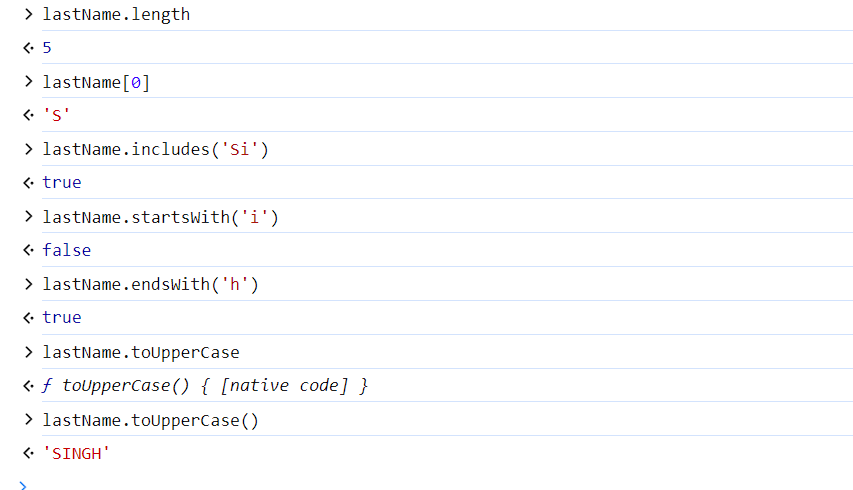
math.random() – will generate random number between 0 to 1.

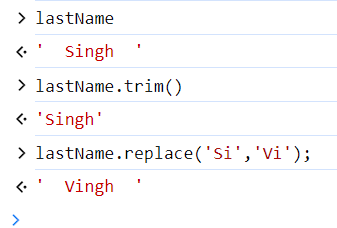
**String() -> In JS string is of two types one is primitive and second is object**

let lastName=new String(‘Avi’); -> it will create object string.

// lastName.

// using dot notation we can treat primitive type into String Object (Js internally treat lastName as String object)

****



**Template Literal** -> we used back-tick character here to show exactly show on the console how we write on an editor.

When we want to set any value in function we can use setter function, and if we want to get any value from function, we can use getter function.

**Arrays – Array is an object or reference type and it is collection of different types of items.**

* Adding new element
* Finding element
* Removing Element
* Splitting Element
* Combining Element

Mero ko ek method execute karna hai, par us method ko execute krne se pehle mujhe kuch data ya value chaheye jo mujhe ek function/method se mile ga, jis function se mujhe data mile ga use mai callback function bol raha hnnnn……

Aapko koi action krna hai par us action krne ke liye aap kise or action par dependent ho.

**Comparison function/Predicate function/Callback function - > A callback function is a function passed into another function as an argument which is then invoked inside the outer function to complete some kind of action.** (50:00)

// here we are searching in object using find() method

// arrayName.find(callback funtion) -> it will return object

// Predicate function means on which condition we are going to find object

// let course = courses.find(function(course){

//     return course.name=='Avi';

// });

// console.log(course);

// we can deduce the upper callback function using arrow function (More READABLE)

let course=courses.find(course => course.name == 'Rahul');

console.log(course);

**JS-4**

// Function -> a block of code that fulfils a specific task

// Hoisting is the process of moving function declaration to the top of file, this is automatically by JS engine

Hoisting only shift function declaration not functions assignment.

// spread operator is used for concatenating and copy of array...

// function sum(a,b){

//     // JS has special object called 'arguments' which store all the data

//     // console.log(arguments);

//     let total=0;

//     for(let value of arguments)

//         total=total+value;

//     return total;

// };

// // funtion is also behaves as dynamic

// // console.log(sum(1,2));

// // console.log(sum(1));            // 1+undefined=NaN (not a number)

// // console.log(sum());

// // console.log(sum(1,2,3,4,5,6));

// console.log(sum(1,2,3,4,5,6));   // now we can pass any number of arguments or parameter.

// getter to fetch the value and setter to set the value

**JS DOM 1**

**Window**- window is a **global object** which is created by browser. It represents or control a browser window. In window object DOM, BOM and JS core functionality (Arrays, function and object) lies here. It is a top-level entity. We can control browser window by window global object.

**DOM**- Document object model – in this we convert all the html code into a JS object that we called document. It is tree like structure. Example – document.body

**BOM –** it allows JS to talk to browser about matter (location or history etc.) other than content of page. Because content of page is handled by html.

* Character is converted into tag then tag is converted into token with the help of tokenizer, then token is converted into nodes then last in DOM.

getElementById(‘id’) – it is used to fetch the element whose id is passed in an input parameter, it is always called on document object model. It returns a single object because id is unique.

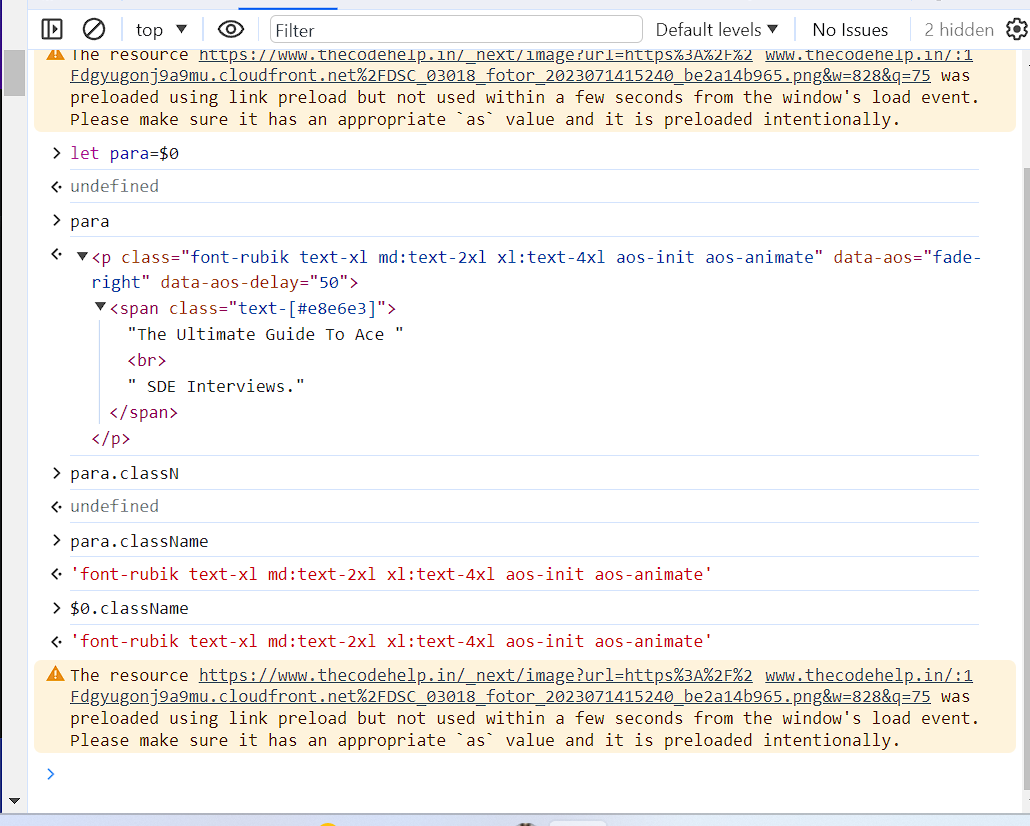
For multiple object we can call getElementsByClassName(‘class’) because class can be multiple.

And same with getElementsByTagName(‘p’) -> it will also return multiple objects.

Prototype (return type) for above two method is HTML collection (Array-like-object).

BOTH METHOD USE DOCUMENT OBEJCT, THE LIST RETURN IS NOT AN ARRAY AND BOTH RETURN MULTIPLE ITEMS. (THINKS TO KEEP IN MIND)

IF WE SELECT ELEMENT IN ELEMENT SECTION AND WRITE ON CONSOLE $0, THEN THAT ELEMENT GOT SELECTED.



We can use querySelector() method to select element just like we do in CSS. We used query selector method and pass it to a string just like that CSS selector. It will return single object (first-one). For multiple object return use querySelectorAll().

querySelector(‘#header’)

querySelector(‘.header’)

querySelector(‘header’)

**UPDATE EXISTING CONTENT:-**

* innerHTML-> get an element and all of its descendent elements and set an element’s HTML content.
* outerHTML
* textContent
* innerText

the difference between textContent and innerText is that in textContent show all the content irrespective of those elements whose display is hidden but in innerText show only those content whose display is not hidden. (41:32 JS 1)

the difference between innerHTML and textContent is innerHTML try to render those tag which is written inside it but in textContent those tags were treat as a text.

The [Element](https://developer.mozilla.org/en-US/docs/Web/API/Element) property **innerHTML** gets or sets the HTML content within the element.

The innerText/textContent property sets or returns the text content of an HTML element. While this treat as a normal text.

The innerHTML property sets or returns the HTML content (inner HTML) of an element. This treats the value as the html syntax.

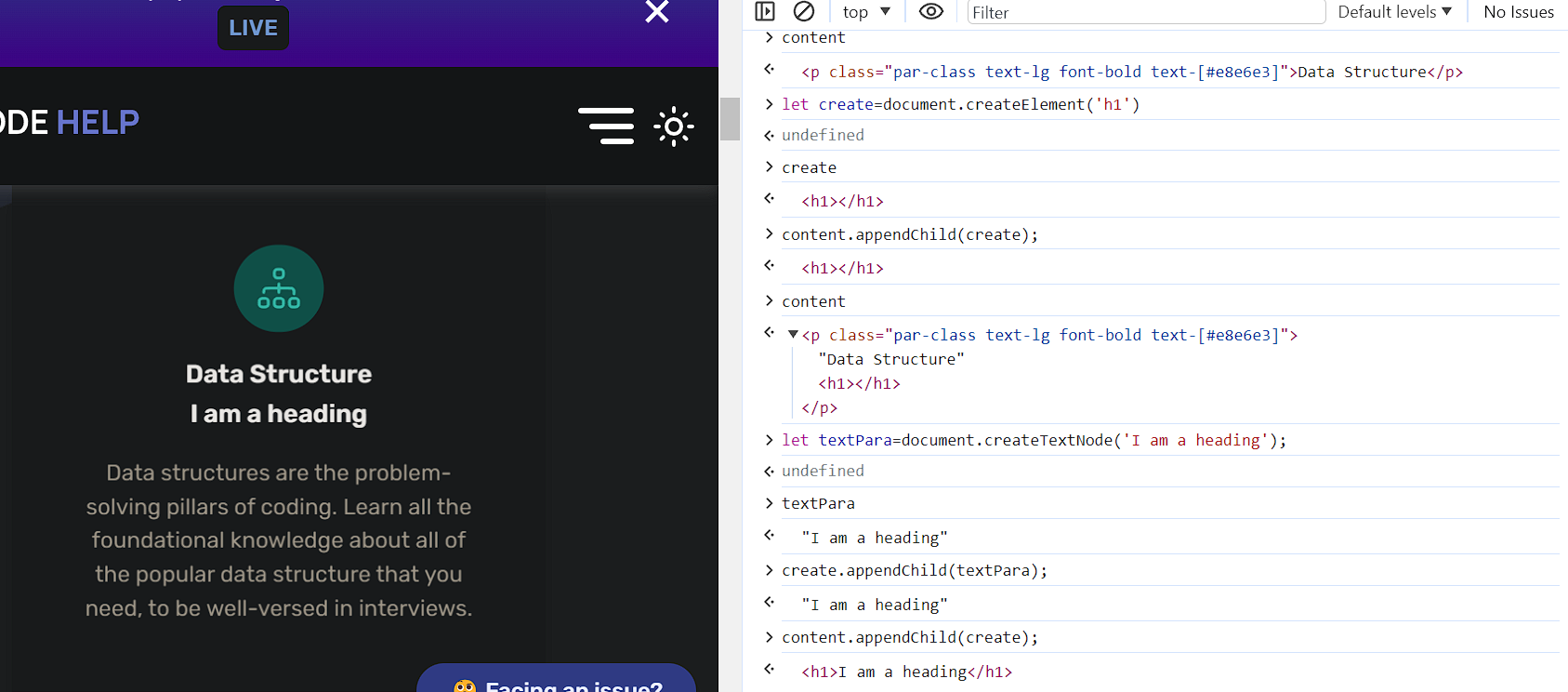
The value property sets or returns the value of the value attribute of a text field.

**ADDING NEW ELEMENT/CONTENT USING JS**

createElement(); method is use to create or add html element using JS and by using appendChild() method we can append or add that html element at the end.



**CREATION OF TEXT-NODE ->**  In that h1 tag if we want add any heading using createTextNode(‘’)



**Easy Way:**

**Let myHead=document.createElement(‘h1’);**

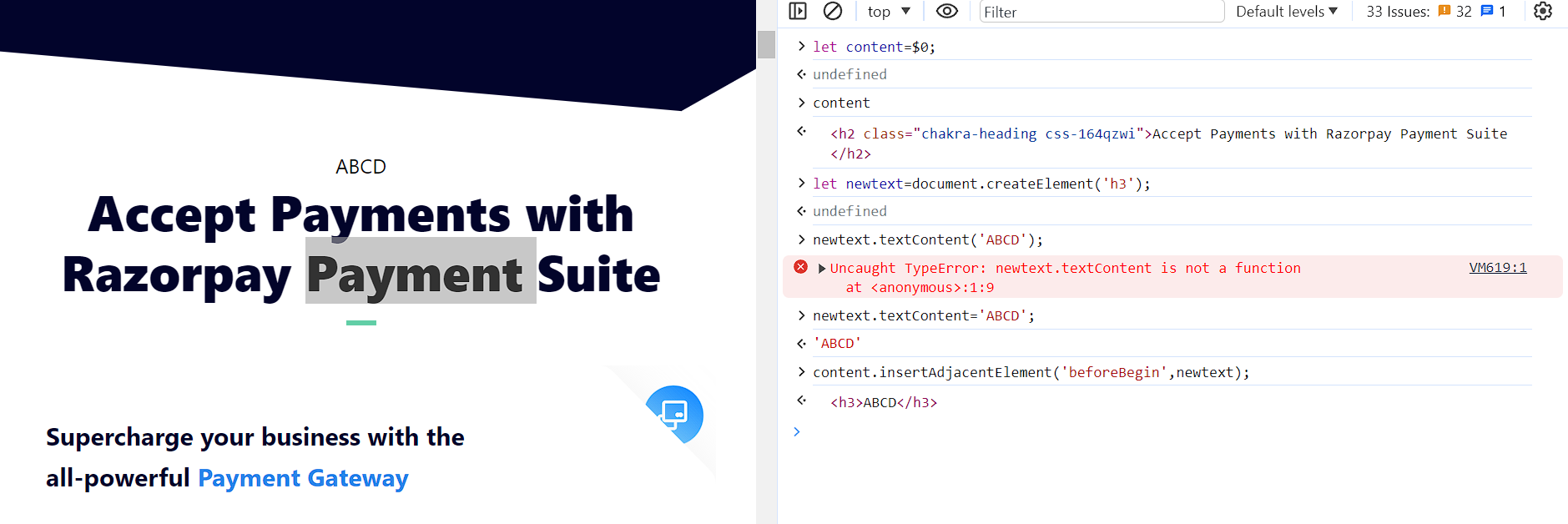
**myHead.textContent=”I am a heading”; // myHead.innerText=”I am a Heading”;**

**content.appendChild(myHead); // this will add at last sibling.**

**But if we want to add at our position??? Then we use insertAdjacentHTML() to position newly created element.**

**Point to Remember->**

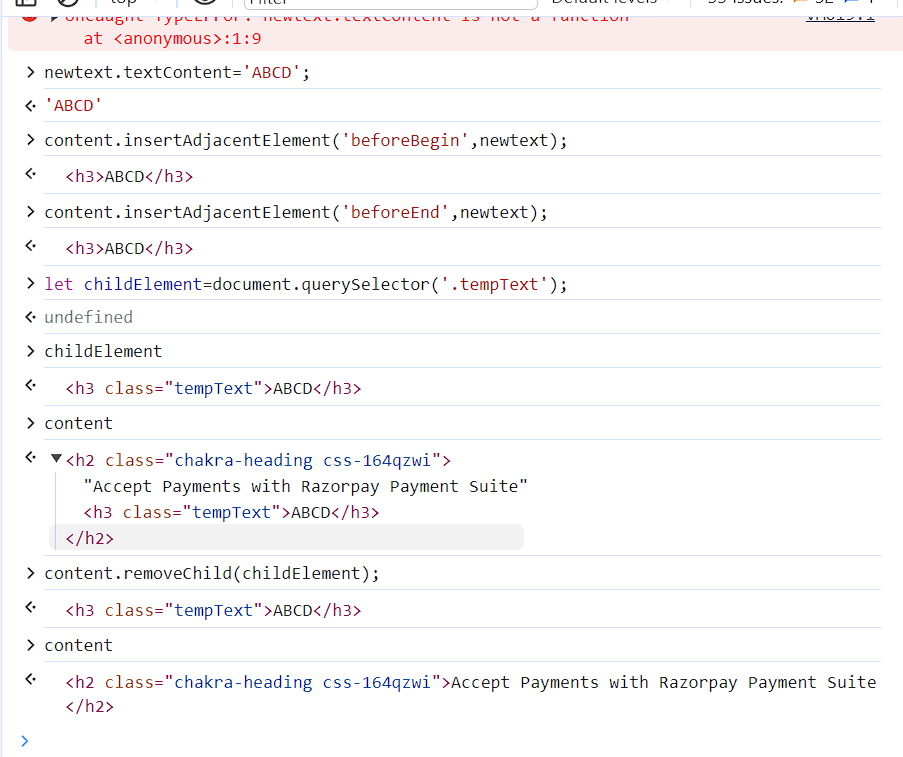
* it has to be called 2 with arguments first is location/position and HTML text and 2nd is HTML content/text to be inserted.
* There is 4 position or location that we offer beforebegin, afterbegin beforend, afterend.
* beforebegin insert html text as previous sibling.

****

**HOW TO REMOVE AN HTML ELEMENT?**

By using removeChild() -> it is opposite of appendChild(), a condition is a parent and child element to be known.

**SYNTAX** -> parent.removeChid(childElement);



There is also method in which we don’t have to know the value of parent.

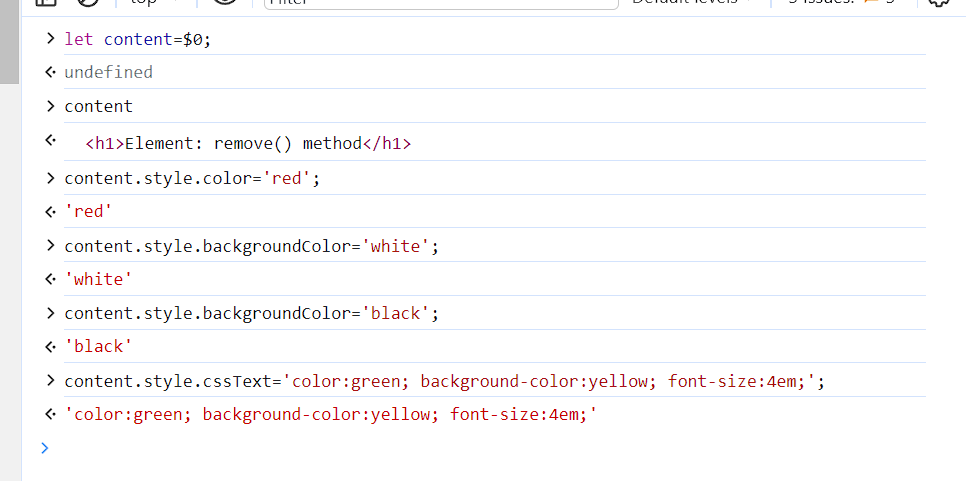
parent=childElement.parent // this will find parent // -> childElement.parent.remove(childElement)

parent.remove(childElement)

**CSS**

If we want to style our page content using JS then there are five methods

1. .style -> at a time we can change single property (DRAWBACK)
2. .cssText -> but in this case we can change multiple property at a time.
3. .setAttribute -> using this method we can change multiple property at a time and we can add or set any attribute like id and class of HTML element. (Here we are breaking separation of concern)
4. .className -> gives a string of class name
5. .classList -> return an array of classes(Object), in this there are four methods add(), remove(),toggle() and contains().



With DOM, we can easily access and manipulate tags, IDs, classes, Attributes, or Elements of HTML using commands or methods provided by the Document object. Using DOM, the JavaScript gets access to HTML as well as CSS of the web page and can also add behavior to the HTML elements.

**JS DOM 2**

**Browser Events – ‘click and double click, scroll’ and action is defined by listener.**

**Event are basically an announcement which are triggered or throw by the browser.**

**Classes are blueprint and object are reality and in JS interface is a blueprint.**

**Event target :- is an interface implemented by object that can receive events and may have listeners(After receiving that event how to respond) for them. Interface basically means blueprint.**

**Event target have 3 methods -> addEventListener, removeEventListener and dispatchEvent().**

**Event target is top level entity. (Then iske andar documents,paragraph,article and video come under this)**

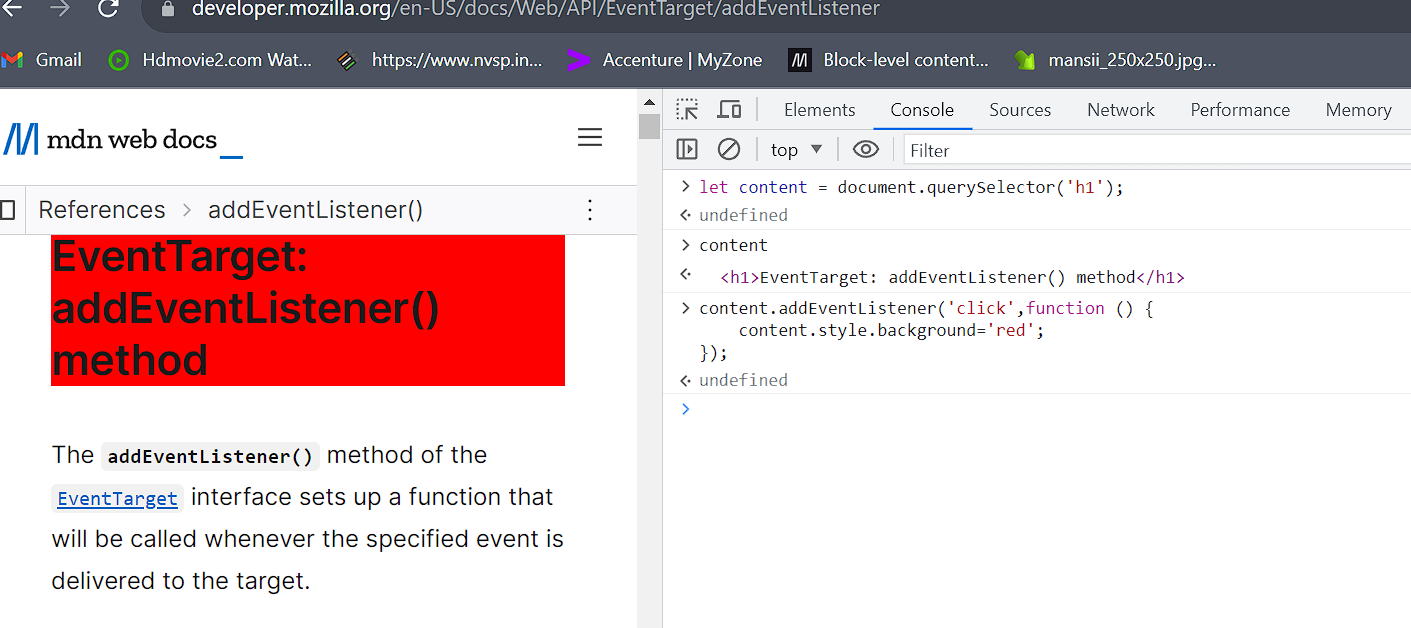
**Nodes inherit from eventTarget means those properties and method can exist in a eventTarget can also exist in a Nodes.**

**And Element are inherited by nodes. (So Element have both properties and method of eventTarget and Nodes)**

**Event hone ke baad jo action mai lunga wah kaun define krta hai wah event listener define krta hai.**

**Kise event par hum agar action perform karna chahete hai to event listener ka use krte hai.**

**addEventListener basically means listen to event or respond to event or hook into event.**

****

monitorEvents(document) -> This method will let us see different event as they occurring.

unmonitorEvents(document) -> this method is used turn off the events.

**SYNTAX -> eventTarget.addEventListener(<event-to-listen-for> , <function to run where event happened>)**

EventTarget means on which component you want to apply event listener for example document,paragraph,heading,div,video,article etc…

Event type -> click,double-click, scroll, DOM CONTENT LOADED, LOAD

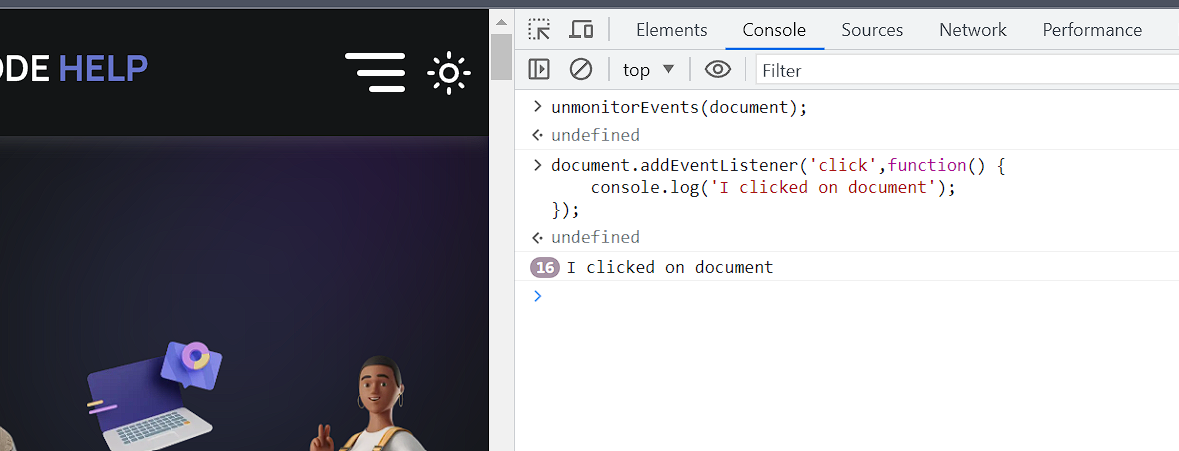
Function -> define what to do when event occur.

Loose Equity allows **Type** **Coercion** means where JS will try to convert the items being compared to same type.

Strick Equity prevent from **Type Coercion**

**reomoveEventListener methods require you to pass the same exact listener function to it as the one you have passed to addEventListener.**

**To fulfill removeEventListener we have three condition -> same eventTarget, same type(click) and same function.**

****

**Phases of an event –**

1. Capturing phase -> in this phase we find eventTarget
2. At target phase
3. Bubbling phase

By default addEventListener is executed at Bubbling phase.

addEventListener(‘click’,print,true); true value on or execute addEventListener at capturing phase.

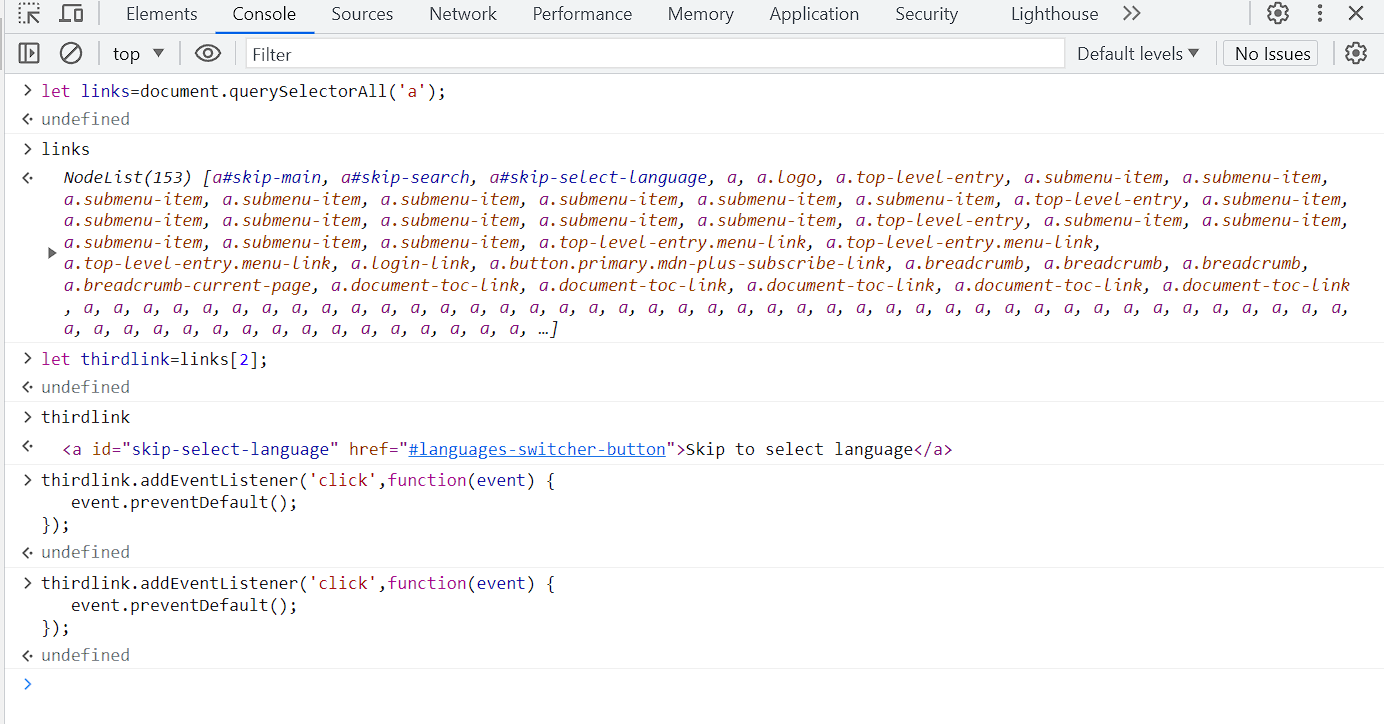
Can we execute event listener at target phase?

**The Event Object –** when an event occur or triggered, addEventListener() (listener) function has get or received event object which has lot of information about event.

We can stop default behavior of an element by using .preventDefault().

preventDefault() method basically used to prevent default action example anchor tag.

Example of preventDefault().



**How to avoid too many events- ??(Last 20 min) REFER THIS**

**Event Target –** The target property returns the elements where the event occurred.

**NODENAME - >** specific filtering(Listener) element ke beech me kar sakte hai through nodename

**DOM CONTENT LOADED and LOAD event….**

**JS DOM 3 (Revision all previous lecture)… is all about PERFORMANCE.**

**Performance –** how to measure a speed of a code? How to write efficient & performing code

JS provide a standard way to measure how long your code take to run, it provides a method called **performance.now()** which return time stamp.

**REFFLOW- whenever we insert any new element then there is certain mathematical calculation is going around like dimension, positioning. And all the calculation is termed as REFLOW.**

**‘It is the process of calculating the dimension and position of page element. This is computationally intensive task. And repaint is the process of drawing pixel on a screen’**

**After this we got new screen layout, we show our screen layout on display by pixel by pixel is termed as REPAINT.**

Repaint is faster than reflow….

**Best practice to reduce the number of reflow and repaint by Document Fragment.**

**Document Fragment-** it is light-weight document object, addition of any new element do not require any reflow and repaint.

So, add your new element say ‘p’ into document fragment and at the end add in document.

If we want to update multiple times DOM, we use document fragment.

**The Call Stack- A list that can track the multiple function-calls.**

**JS engine keeps the call stack of the function that are running when a function is invoked. It is added to the list when all of the code inside a function has been run then the function is removed from the call stack.**

**Single-threading – one command at a time, JS is single-threading language. (processing of one command at a time).**

**JS run-to-completion nature of code and JS does not execute multiple lines/function at the same time.**

**Event LOOP-**

**Event loop: An event loop is something that pulls stuff out of the queue and places it onto the function execution stack whenever the function stack becomes empty.**

Synchronous means occurring at the same time.

EventListener function is not synchronous or asynchronous as it is depends on out action. (click, double click)

Asynchronous code is using JS event loop.

Any Asynchronous code is handled by browser. And after event is triggered, it goes into queue and execute only if the call stack is empty.

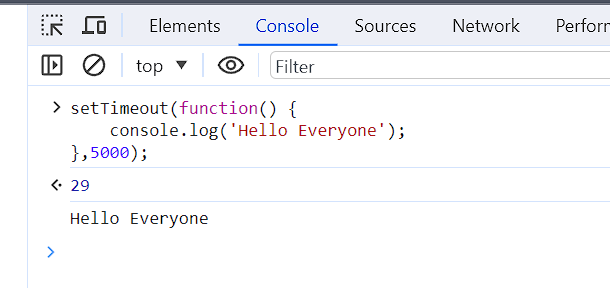
When a-syn code is ready to be executed it is moved to the queue, where it wait until the call stack is empty.

Whenever the call stack is empty, code move from queue to call stack and get executed.

Queue asyn code ko tbhi execute karata hai jab call stack empty hota hai.

**Asyn code runs at later point of time. Not run immediately like setTimeout(), eventListener.**

**setTimeOut() –** this method will run code at a later point in time.



**Asynchronous code will execute but when we don’t know.**

JS is single threading language and concurrency is handled by asynchronous code and asynchronous code is handled by event loop.

**Features of asyn code-**

1. Clean and concise
2. Better error handling
3. Easier debugging.

**Promise-** Anything we want execute parallelly in background in JS using promise.

Promise has two states either it will reject or resolve.

Any value received from fulfilled state is handled by **then()** and rejected state is handled by **catch().**

**Async-wait –** special syntax used to work with multiple promises. If we want to wait promise 2 until first promise get completed so we can use async-wait.

We can make any function async and async function will always **return promise**.

You can wait any promise using await keyword.

We can use **await keyword** in async function.

**Fetch API –** Api is a tool to communicate between two software components.

Through fetch API we can sent or retrieve data from network.

Fetch api will return promise.

**We can put network call into async function().**

**\*Closure – In closure inner function and references of data is stored. Saare nested function ke closure bane ge and surrounding state ke saare function and data ka reference bane ga.**

If we want to change CSS property we have 5 methods -> .style(), .bg(), cssText(), classList(), className(), setAttribute() using JS.

We can use querySelector with class, Id, tag and **custom attribute.**

2 software ke beech ka communication medium (API).