10 May 2024 11:17

DOM + JS mord

window → top level entity
window → global object
- → Created by browser
- → represented in browser
- → DOM, BOM, JS core

DOM → Document object Modal
↳ HTML code convert into JS object

BOM → location, communication, other then content

DOM :-

<html>
<body>...</body>
<head>...</head>
</html>

Character → tokenizer → tags → token → nodes → DOM

Element fatch

document.getElementbyID(' ')
- ID name
- it is called on document object
- it always a single object

getElementsbyclassName(' ')
- class name

getElementsByClassName(' ') ← class name

getElementsByTagName(' ') ← tag name

↑ Multiple item return.

querySelector('#header') ← id

('.header') ← class

('header')

} only one item display.

querySelectorAll(' ') ← Multiple item display

Web page :- update existing context

innerHTML → get/set HTML content

outerHTML

textContent
innerText } → get/set text content

Adding new element

↳ .createElement(' ')

Content.appendChild(newchild name)

↑

any element

Creating Text Node :-

① let newPara = document.createElement('P'); ← last me add

let textPara = document.createTextNode('Meee');

let textpara = document.createTextNode('Meee');

newPara.appendchild(textpara)

② let MyPara = document.CreateElement('P')

MyPara.textcontext = "Meet"

Contat.appendchild(MyPara)

insertAdjacentHTML()

↳ has to be called 2 days

↳ location → where

↳ HTML → what

beforebegin beforeend

afterbegin afterend

Remove

Parent name .removechild(child name)

↳ perent element known

↳ which child is remove that known

child.parent.removeChild(Child name)

CSS proptity change :-

↳ .style

.CSS Text

.SetAttiubet

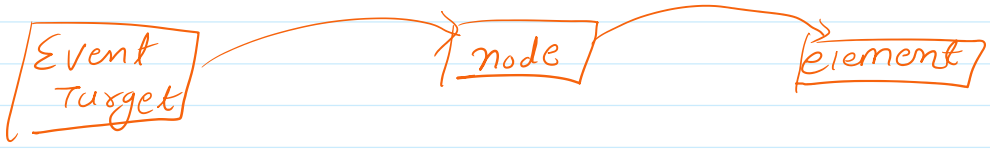
.className

• Classlist

* browser event (Announcement)
  → monitorEvents() → all event saw

EventListener :-

EventTarget :- interface (blueprint) implemented by object that can receive events & may have listener for them.
  → addEventListener();
    removeEventListener();
    dispatchEvent();



addEventListener() → listen to event

pseudocode

<event-target>.addEventListener(<even-type>, function)

document
p
h1
h2, a, vidio

click
double-Click

define what to do when event happen

Remove Event Listener → same target
  → same type
  → same function

function print() {
  console.log("Hi");
}

document.addEventListener('click', print);

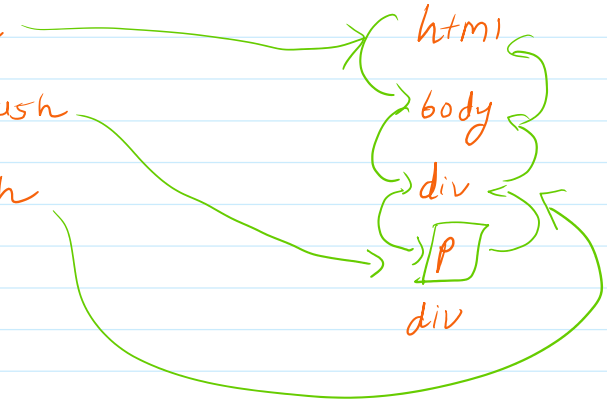
document.removeEventListener('click', print);

phases of an Event

↳ capturing phash

↳ At target phash

↳ Bubbling phash



By default run in Bubbling phash

.addEvent('click', print, true)

↑ captring phash

The Event Object :-

↳ when event occur, .addEventLister() function

↳ more information

The default action :-

anchor tag :- link open

.preventDefault()

↑ close kar shakate hai default action

Avoid to many event

Performance

Synchronce :- occuring at a same time

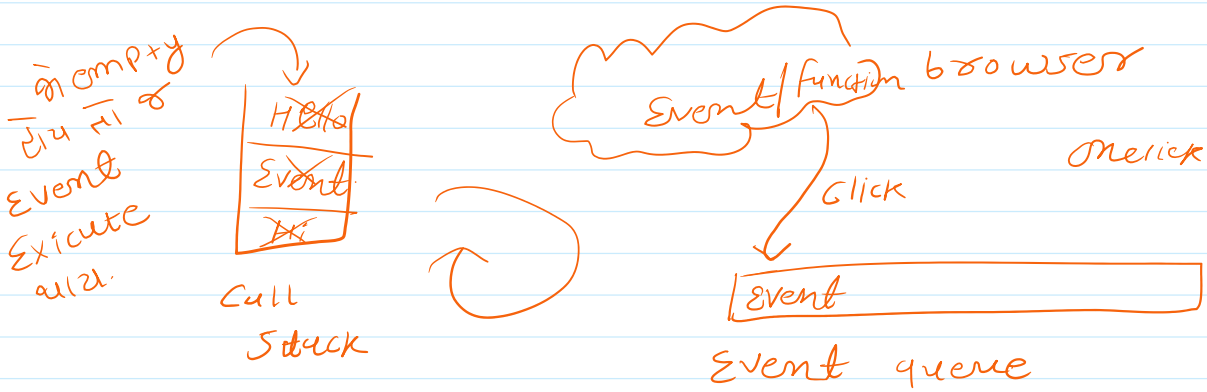
function a() {

≡ } all line one by one exicuted.

}

not - synch = EventListner

* Event Loop



①

Console.log('Hi');

Eventlister

Console.log('Hello');

setTimeout()

setTimeout(function(){

console.log('Hi');

}, 4000);

minmum time.
but call stack is empty
then run.

→ API :- application programming Interface.

featurs of Asyc code
- ↳ better error handling
- → Easy debugging.

Promise
- ↳ parall exitucuted (background)

call back function

```
let a = new Promise(function(resolve, reject){

})
```

promise
- → fullfill
- → Reject
- → then()
- → catch() ← error.

Async - await → spcial syntex used to work with promies

↓

rukana

fetch API → fetch(url);

↳ send