**JAVASCRIPT EVENTS**

1. **WHAT IS EVENT IN JS.?**

The change in the state of an object(like button,) is known as an Event. Events are fired to notify code of "interesting changes" that may affect code execution.there are various events which represents that some activity is performed by the user or by the browser

1. **WHAT IS EVENT HANDELER?**

When java script code is included in HTML, jS react over these events and allow the execution. This process of reacting over the events is called Event Handling.Thus, j.s handles the HTML events via Event Handlers.

**3. Some of the type of events.**

* 1. Mouse Event(Click, Double click etc)
* 2. Keyboard Events(Key-press, Key-up, Key-down)
* 3. Form Event(Submit)
* 4. print event and Many more..

**4. Way of Event handling**

* Inline Event handling ( Write event handlers inside the HTML tag )

Disadvantages => This way is effected our code readability

* External Event handling( Write event handlers inside j.s file with Arrow function)

Disadvantages =>in this way we can not write multiple function for same event. if you write multiple function for same event in that case override the function. and execute second event.

* Event Listeners : Event listener is function that is always listen the event .means it always ready to perform the action when any event occur.  it is better way to handle the event . we can create multiple event listener for one event

**5.What is Event object**

It is a special object that has details about the event.

like

* type of Event(Means that event is mouse event, Pointer event, Keybord Event)
* target of Event(Means that event occur on div or input or button)
* All events handlers have access to the event objects properties and method

****6.What is “propagation”?****

Propagation refers to how events travel through the Document Object Model (DOM) tree. The DOM tree is the structure which contains parent/child/sibling elements in relation to each other. You can think of propagation as electricity running through a wire, until it reaches its destination. The event needs to pass through every node on the DOM until it reaches the end, or if it is forcibly stopped.

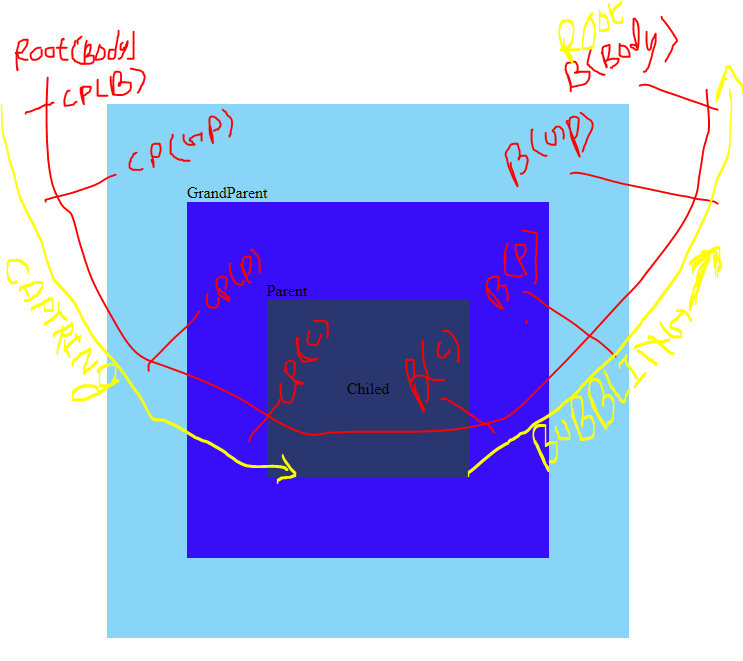
1. ****Event Bubbling and Capturing****

Bubbling and Capturing are the two phases of propagation. In their simplest definitions, ****bubbling**** travels from the target to the root, and ****capturing**** travels from the root to the target. However, that doesn’t make much sense without first defining what a target and a root is.

Other

**Event Bubbling : it is process of automatically call back event . means suppose you create three boxes and all boxes**

**lies inside  the another boxes and if you click on any inside box in that case automatically call the event of upper box.**



# **Event delegation**

Capturing and bubbling allow us to implement one of the most powerful event handling patterns called event delegation.

The idea is that if we have a lot of elements handled in a similar way, then instead of assigning a handler to each of them – we put a single handler on their common ancestor.

In the handler we get event.target to see where the event actually happened and handle it.

const childBox=document.querySelector(".child");

const parentBox=document.querySelector(".parent");

const grandParentBox=document.querySelector(".grandParent");

const body=document.querySelector("body");

// Capturing/////////////////////////////////////////////////

childBox.addEventListener("click",()=>{

    console.log("Cpturing =>  Child Box ");

},true)

parentBox.addEventListener("click",()=>{

    console.log("Cpturing =>  Parent Box ");

},true)

grandParentBox.addEventListener("click",()=>{

    console.log("Cpturing =>  Grand Parent Box ");

},true)

body.addEventListener("click",()=>{

    console.log("Cpturing =>  body");

},true)

// Bubbling/////////////////////////////////////////////////////

childBox.addEventListener("click",()=>{

    console.log("Bubling =>Child Box ");

})

parentBox.addEventListener("click",()=>{

    console.log("Bubling Parent Box ");

})

grandParentBox.addEventListener("click",()=>{

    console.log("Bubling Grand Parent Box ");

})

body.addEventListener("click",()=>{

    console.log("Bubling body");

})

//7.Event delegation//////////////////////////////////////////

grandParentBox.addEventListener("click", (e)=>{

    console.log(e.target)

})