@startwithmani @M.serisha Kothapalli

Why 3 ways of Event handling in JS?

swipe

```
<button onclick="btnClick()">Click Me</button>
<script>
  function btnClick() {
    console.log("Button Clicked")
  }
</script>
```

1

```
<button id="myBtn">Click Me</button>

<script>
  var btn = document.getElementById("myBtn");
  btn.onclick = btnClick

  function btnClick() {
    console.log("Button Clicked")
  }
</script>
```

2

```
<button id="myBtn">Click Me</button>

<script>
  function btnClick() {
    console.log("Button Clicked")
  }

  var btn = document.getElementById("myBtn")
  btn.addEventListener("click", btnClick, false)
</script>
```

3

swipe

Quick Intro To Events

An event is an **action** that occurs as per the user's instruction as input. **JavaScript's interaction** with **HTML** is handled through events.

We have various kinds of events like

1. Window/Document events
2. Mouse / Keyboard events
3. Window/Document events etc....

Event Handling??

This is fine but we need to somehow react to these events.

This process of **reacting** is called event handling.

swipe

Have you ever tried to figure out what exactly is the difference between the three code snippets ?? 🤔

Why do we have **three** ways to **handle events**? Isn't it one would be enough to handle?

Below are the terms that we use for different ways of handling events based on previous slide code snippets (1,2,3)

1. HTML event handlers
2. DOM level event handlers
3. Event Listeners

swipe

1 HTML event Handlers

Code 1 is an example of an inline event. Here the event is specified with a function as an attribute to the **HTML tag.**

Assigning event handlers using HTML event handler attributes is considered as bad practices.Try to avoid using this approach as much as you can.. WHY???

❌ **Readability :** The event handler code mixed with the HTML code,turns out difficult to read when we have multple event handlers on same element.

❌ **Timing issue :** If the element is loaded fully before the JavaScript code, users can start interacting with the element on the webpage which in turn gives an error.

swipe

2 DOM level event Handlers

In this case, all you have to do is give your element an identity (which could be most preferably, id, or class - not necessarily).

✓ One main advantage of this method when compared to previous one (HTML event handlers) is that, the scope of the function can easily be controlled.

So Code 1 & 2 (Event handlers) almost do same thing.

One **important common point** for both of them is that For a given element, you can only have **one event handler** per **event type**......

Meaning if you add two event handlers for the same element on same event , the second event handler will **overwrite** the first and only that event will trigger.

swipe

3 Event Listeners

The other method to use events in JavaScript is by adding an event listener to an object. By adding an event listener to an object, we can catch a wide range of events triggered by the user or the browser.

The **addEventListener( )** method specifies a function that will be invoked when the given event is listened on the element.

- ✓ Event propagation can be controlled using **useCapture**
- ✓ One more over the above two methods is that, it can have multiple event handlers applied to the same element. It doesn’t overwrite other event handlers.

swipe

Did you find it helpful??

♡ Like this post!

Share with your friends

Save it for later

Follow for more!

@startwithmani

@M.serisha Kothapalli