JS

JavaScript Class

#JavaScript Notes

Storage

JS

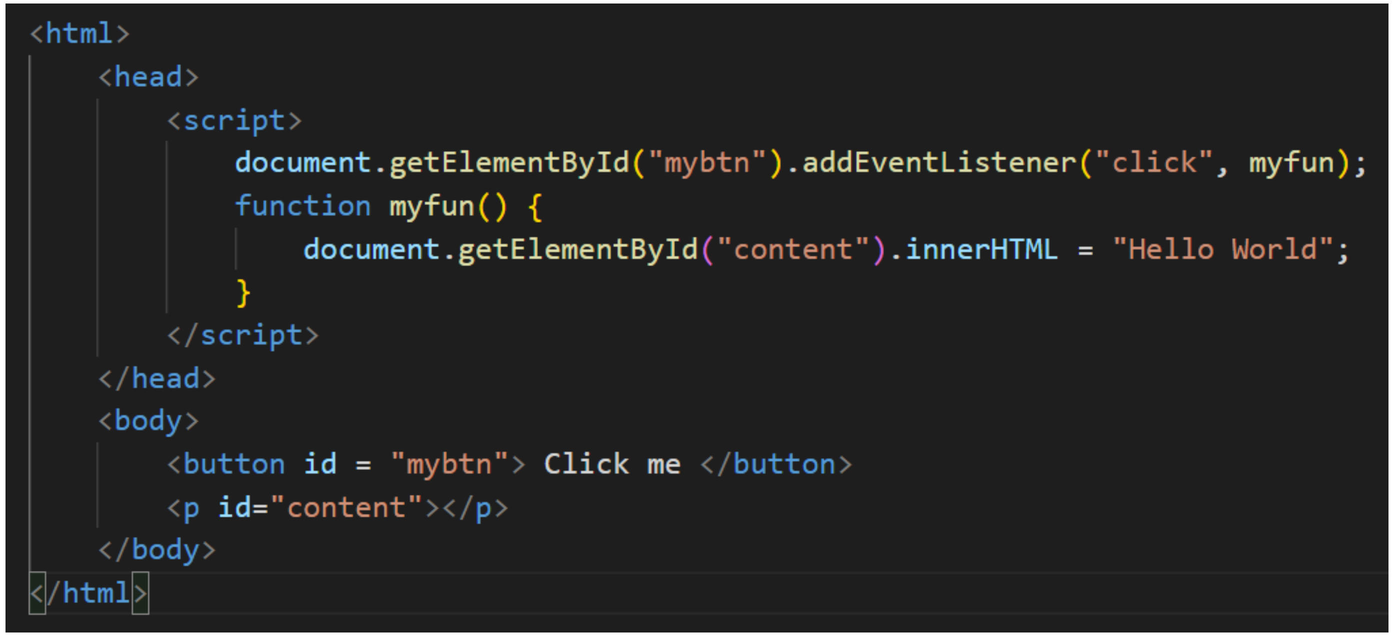
What is Event Listener?

- An event listener is a procedure in JavaScript that waits for an event to occur. The simple example of an event is a user clicking the mouse or pressing a key on the keyboard.
- The addEventListener() is an inbuilt JavaScript function which takes the event to listen for, and a second argument to be called whenever the described event gets fired.
- Any number of event handlers can be added to a single element without overwriting existing event handlers.

addEventListener() in JavaScript

- The addEventListener() method attaches an event handler to the specified element.
- This method attaches an event handler to an element without overwriting existing event handlers.
- You can add many event handlers to one element.
- You can add many event handlers of the same type to one element, i.e two “click” events.
- Event listeners can be added to any DOM object not only HTML elements. i.e the window object.
- The addEventListener() method makes it easier to control how the event reacts to bubbling.

- element.addEventListener(event, function, useCapture);
- function: It is also a required parameter. It is a JavaScript function which responds to the event occur.
- useCapture: It is an optional parameter. It is a Boolean type value that specifies whether the event is executed in the bubbling or capturing phase.

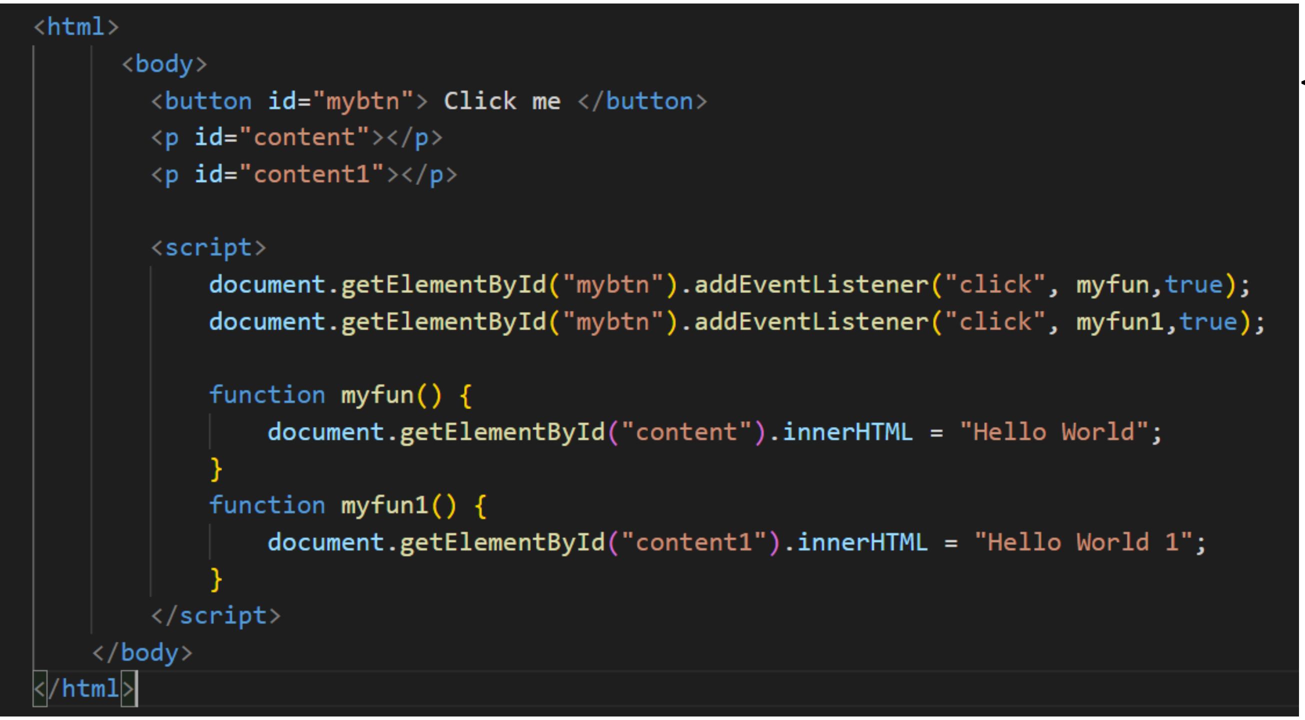

```
<html>
    <head>
        <script>
            document.getElementById("mybtn").addEventListener("click", myfun);
            function myfun() {
                document.getElementById("content").innerHTML = "Hello World";
            }
        </script>
    </head>
    <body>
        <button id = "mybtn"> Click me </button>
        <p id="content"></p>
    </body>
</html>
```

```
<html>
   <head>
      <script>
         document.getElementById("mybtn").addEventListener("click", myfun);
         function myfun() {
            document.getElementById("content").innerHTML = "Hello World";
         }
      </script>
   </head>
   <body>
      <button id = "mybtn"> Click me </button>
      <p id="content"></p>
   </body>
</html>
```

Add Many Event Handlers to the Same Element

- The JavaScript addEventListener() method lets you add multiple event handlers to a single element

- document.getElementById("mybtn").addEventListener("click", myfun);
- document.getElementById("mybtn").addEventListener("click", myfun1);


```
<html>
    <body>
      <button id="mybtn"> Click me </button>
      <p id="content"></p>
      <p id="content1"></p>

      <script>
          document.getElementById("mybtn").addEventListener("click", myfun,true);
          document.getElementById("mybtn").addEventListener("click", myfun1,true);

          function myfun() {
              document.getElementById("content").innerHTML = "Hello World";
          }
          function myfun1() {
              document.getElementById("content1").innerHTML = "Hello World 1";
          }
      </script>
  </body>
</html>
```

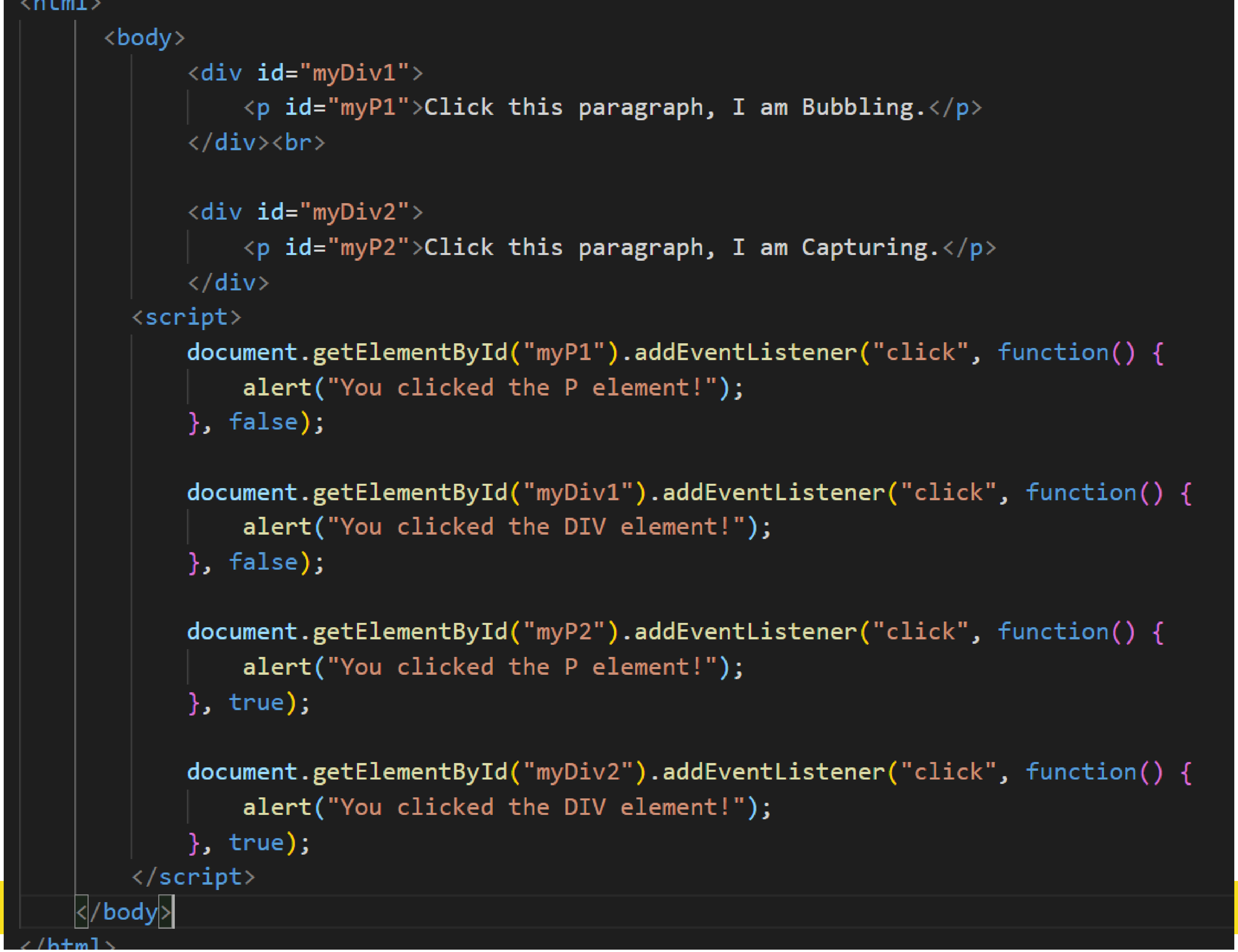
```
<html>
    <body>
      <button id="mybtn"> Click me </button>
      <p id="content"></p>
      <p id="content1"></p>

      <script>
        document.getElementById("mybtn").addEventListener("click", myfun,true);
        document.getElementById("mybtn").addEventListener("click", myfun1,true);

        function myfun() {
          document.getElementById("content").innerHTML = "Hello World";
        }
        function myfun1() {
          document.getElementById("content1").innerHTML = "Hello World 1";
        }
      </script>
    </body>
</html>
```

Event Bubbling or Event Capturing

- Event propagation defines the element order when an event occurs. For example, when you have an <img> element in a <div>, and the <img> element is clicked, which click event will have to be handled first?
- In the case of bubbling, the element that is on the lowest level event is handled first, and the outer ones afterwards. For example, the click event on <img>, and the click event on <div> after.
- This order is reversed in the case of capturing the click event on <div> is handled first, then the click event on <img>.
- When using the JavaScript addEventListener() method you may set which propagation method will be used with the useCapture parameter.
- By default, this parameter is set to false, meaning that bubbling will be used, and only uses capturing if this value is manually set to true.

Removing Event Handlers

```
target.removeEventListener(event, function, useCapture);
```



```
<html>
    <body>
        <div id="myDiv1">
            <p id="myP1">Click this paragraph, I am Bubbling.</p>
        </div><br>

        <div id="myDiv2">
            <p id="myP2">Click this paragraph, I am Capturing.</p>
        </div>
      <script>
        document.getElementById("myP1").addEventListener("click", function() {
            alert("You clicked the P element!");
        }, false);

        document.getElementById("myDiv1").addEventListener("click", function() {
            alert("You clicked the DIV element!");
        }, false);

        document.getElementById("myP2").addEventListener("click", function() {
            alert("You clicked the P element!");
        }, true);

        document.getElementById("myDiv2").addEventListener("click", function() {
            alert("You clicked the DIV element!");
        }, true);
      </script>
    </body>
</html>
```

JS

```
<html>
   <body>
      <div id="myDiv1">
        <p id="myP1">Click this paragraph, I am Bubbling.</p>
      </div><br>

      <div id="myDiv2">
        <p id="myP2">Click this paragraph, I am Capturing.</p>
      </div>
    <script>
      document.getElementById("myP1").addEventListener("click", function() {
        alert("You clicked the P element!");
      }, false);

      document.getElementById("myDiv1").addEventListener("click", function() {
        alert("You clicked the DIV element!");
      }, false);

      document.getElementById("myP2").addEventListener("click", function() {
        alert("You clicked the P element!");
      }, true);

      document.getElementById("myDiv2").addEventListener("click", function() {
        alert("You clicked the DIV element!");
      }, true);
    </script>
  </body>
</html>
```

Timer Functions

JS

Timer

- JavaScript timing events means running the code in defined time intervals.
- You can use JavaScript setTimeOut() function to execute some functionality after a specified amount of time.
- You can use setInterval() function to execute some functionality continuously with defined breaks inbetween.
- You can cancel a setTimeOut() by calling clearTimeOut() function.
- You can cancel a setInterval() by calling JavaScript clearInterval() function.
- You can combine this functionality with other JavaScript features, like a JavaScript alert popup.

setTimeOut()

- it calls a function after a time you have specified passes.
- If you click a button, a JavaScript alert message will pop up after 2 seconds (2000 milliseconds):

```
<button onclick="setTimeout(showAlert, 2000)">Click me!</button>
```

clearTimeOut()

- This JavaScript timer function cancels the JavaScript setTimeout() function before it is executed.
- setTimeout() function again calls a JavaScript alert to pop up after 2 seconds. However, if you call a clearTimeout() function before the seconds pass, it will be canceled:

```
<button onclick="myVar = setTimeout(showAlert, 2000)">Try it</button>

<button onclick="clearTimeout(myVar)">Stop it</button>
```

setInterval()

- This JavaScript timer function sets an interval in milliseconds when something should change. In the example below, the displayed time changes every 2 seconds:

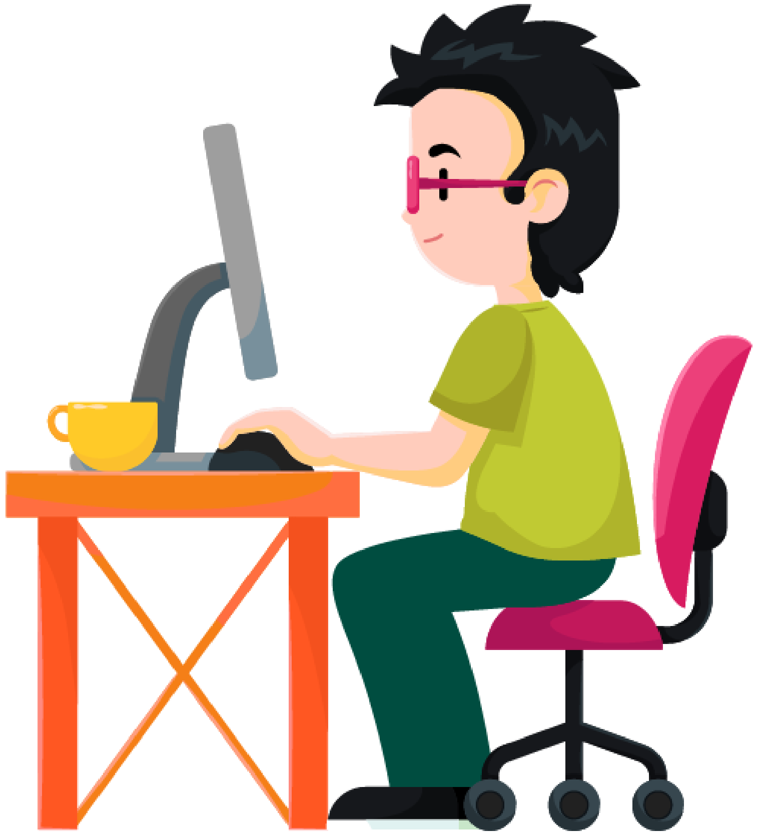
```
var exampleVar = setInterval(exampleTimer, 2000);
function exampleTimer() {
  var d = new Date();
  document.getElementById("example").innerHTML = d.toLocaleTimeString();
}
```

clearInterval()

- This JavaScript timer function clears the interval, stopping it from running:

```
var exampleVar = setInterval(exampleTimer, 0);
function exampleTimer() {
    var date = new Date();
    document.getElementById("example").innerHTML = date.toLocaleTimeString();
}
```

Get Exclusive Video Tutorials



www.aptutorials.com

https://www.youtube.com/user/Akashtips

Get More Details

www.akashsir.com

If You Liked It !

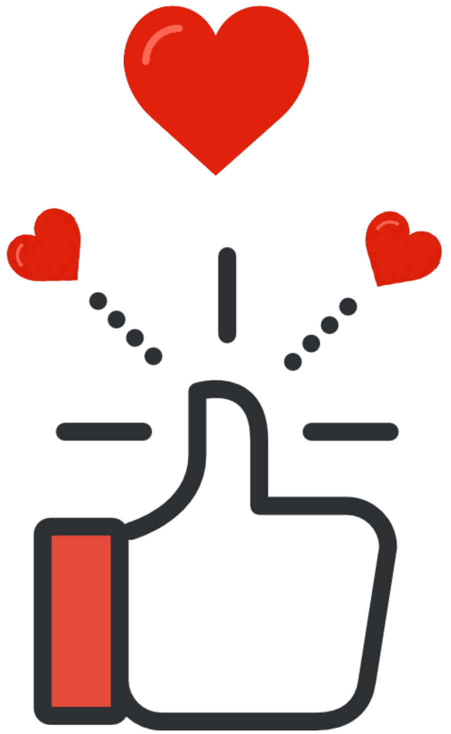
Rating Us Now

Just Dial

https://www.justdial.com/Ahmedabad/Akash-Technolabs-Navrangpura-Bus-Stop-Navrangpura/079PXX79-XX79-170615221520-S5C4_BZDET

Sulekha

https://www.sulekha.com/akash-technolabs-navrangpura-ahmedabad-contact-address/ahmedabad



Connect With Me

Akash Padhiyar
#AkashSir

www.akashsir.com
www.akashtechnolabs.com
www.akashpadhiyar.com
www.aptutorials.com

# Social Info

Akash.padhiyar

Akashpadhiyar

Akash_padhiyar

+91 99786-21654

#Akashpadhiyar
#aptutorials