JavaScript HTML DOM EventListener

❮ Previous Next ❯

The addEventListener() method

Example

Add an event listener that fires when a user clicks a button:

```
document.getElementById("myBtn").addEventListener("click",
displayDate);
```

Try it Yourself »

The `addEventListener()` method attaches an event handler to the specified element.

The `addEventListener()` 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.

You can add event listeners 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.

When using the `addEventListener()` method, the JavaScript is separated from the HTML markup, for better readability and allows you to add event listeners even when you do not control the HTML markup.

You can easily remove an event listener by using the `removeEventListener()` method.

Syntax

```
element.addEventListener(event, function, useCapture);
```

The first parameter is the type of the event (like " `click` " or " `mousedown` " or any other HTML DOM Event.)

The second parameter is the function we want to call when the event occurs.

The third parameter is a boolean value specifying whether to use event bubbling or event capturing. This parameter is optional.

Note that you don't use the "on" prefix for the event; use " `click` " instead of " `onclick` ".

Add an Event Handler to an Element

Example

Alert "Hello World!" when the user clicks on an element:

```
element.addEventListener("click", function(){ alert("Hello
World!"); });
```

Try it Yourself »

You can also refer to an external "named" function:

Example

Alert "Hello World!" when the user clicks on an element:

```
element.addEventListener("click", myFunction);

function myFunction() {
  alert ("Hello World!");
}
```

Try it Yourself »

Add Many Event Handlers to the Same Element

The `addEventListener()` method allows you to add many events to the same element, without overwriting existing events:

Example

```
element.addEventListener("click", myFunction);
element.addEventListener("click", mySecondFunction);
```

Try it Yourself »

You can add events of different types to the same element:

Example

```
element.addEventListener("mouseover", myFunction);
element.addEventListener("click", mySecondFunction);
element.addEventListener("mouseout", myThirdFunction);
```

Try it Yourself »

Add an Event Handler to the window Object

The `addEventListener()` method allows you to add event listeners on any HTML DOM object such as HTML elements, the HTML document, the window object, or other objects that support events, like the `xmlHttpRequest` object.

Example

Add an event listener that fires when a user resizes the window:

```
window.addEventListener("resize", function(){
  document.getElementById("demo").innerHTML = sometext;
});
```

Try it Yourself »

Passing Parameters

When passing parameter values, use an "anonymous function" that calls the specified function with the parameters:

Example

```
element.addEventListener("click", function(){ myFunction(p1, p2);
});
```

Try it Yourself »

Event Bubbling or Event Capturing?

There are two ways of event propagation in the HTML DOM, bubbling and capturing.

Event propagation is a way of defining the element order when an event occurs. If you have a <p> element inside a <div> element, and the user clicks on the <p> element, which element's "click" event should be handled first?

In *bubbling* the inner most element's event is handled first and then the outer: the <p> element's click event is handled first, then the <div> element's click event.

In *capturing* the outer most element's event is handled first and then the inner: the <div> element's click event will be handled first, then the <p> element's click event.

With the addEventListener() method you can specify the propagation type by using the "useCapture" parameter:

```
addEventListener(event, function, useCapture);
```

The default value is false, which will use the bubbling propagation, when the value is set to true, the event uses the capturing propagation.

Example

```
document.getElementById("myP").addEventListener("click",
myFunction, true);
document.getElementById("myDiv").addEventListener("click",
myFunction, true);
```

Try it Yourself »

The removeEventListener() method

The `removeEventListener()` method removes event handlers that have been attached with the addEventListener() method:

Example

```
element.removeEventListener("mousemove", myFunction);
```

Try it Yourself »

HTML DOM Event Object Reference

For a list of all HTML DOM events, look at our complete HTML DOM Event Object Reference.

Test Yourself With Exercises

Exercise:

Use the `eventListener` to assign an onclick event to the `<button>` element.

```
<button id="demo"></button>

<script>
document.getElementById("demo").               ("
</script>
```

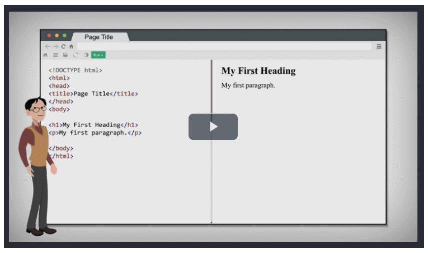
Submit Answer »

Start the Exercise

❮ Previous Next ❯

NEW

We just launched W3Schools videos



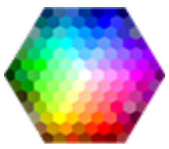
```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

My First Heading

My first paragraph.

Explore now

COLOR PICKER



Get certified
by completing
a course today!

Get started

CODE GAME

Play Game

Report Error Forum About Shop

Top Tutorials

HTML Tutorial
CSS Tutorial
JavaScript Tutorial
How To Tutorial
SQL Tutorial
Python Tutorial
W3.CSS Tutorial
Bootstrap Tutorial

Top References

HTML Reference
CSS Reference
JavaScript Reference
SQL Reference
Python Reference
W3.CSS Reference
Bootstrap Reference
PHP Reference