JS 2018
JS IE / Edge
História JS

JS HTML DOM
DOM Intro
Métodos DOM
Documento DOM
Elementos DOM
HTML DOM
Formulários DOM
DOM CSS
Animações DOM
Eventos DOM
Listener de eventos DOM
Navegação DOM
Nós DOM
Coleções DOM
Listas de nós DOM

JavaScript HTML DOM EventListener

❮ Anterior Próximo ❯

O método addEventListener ()

Exemplo

Adicione um listener de evento que dispara quando um usuário clica em um botão:

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

Tente você mesmo "

O `addEventListener()` método anexa um manipulador de eventos ao elemento especificado.

O `addEventListener()` método anexa um manipulador de eventos a um elemento sem sobrescrever os manipuladores de eventos existentes.

Você pode adicionar muitos manipuladores de eventos a um elemento.

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

O `removeEventListener()` método remove manipuladores de eventos que foram anexados com o método addEventListener ():

Exemplo

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

Tente você mesmo "

Referência de objeto de evento HTML DOM

Para obter uma lista de todos os eventos HTML DOM, consulte nossa Referência de objeto de evento HTML DOM completa .

Teste-se com exercícios

Exercício:

Use o `eventListener` para atribuir um evento onclick ao `<button>` elemento.

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

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

Enviar resposta "

Comece o exercício

❮ Anterior Próximo ❯

Reportar erro Fórum Cerca de Comprar

Principais tutoriais
HTML Tutorial
CSS Tutorial
JavaScript Tutorial
How To Tutorial
SQL Tutorial
Python Tutorial
W3.CSS Tutorial
Bootstrap Tutorial
PHP Tutorial
Java Tutorial
C++ Tutorial
jQuery Tutorial

Top References
HTML Reference
CSS Reference
JavaScript Reference
SQL Reference
Python Reference
W3.CSS Reference
Bootstrap Reference
PHP Reference
HTML Colors
Java Reference
Angular Reference
jQuery Reference

Top Examples
HTML Examples
CSS Examples
JavaScript Examples
How To Examples
SQL Examples
Python Examples
W3.CSS Examples
Bootstrap Examples
PHP Examples
Java Examples
XML Examples
jQuery Examples

Web Courses
HTML Course
CSS Course
JavaScript Course
Front End Course
SQL Course
Python Course
PHP Course
jQuery Course
Java Course
C++ Course
C# Course
XML Course

Get Certified »