How to prevent default event handling in an onclick method?

Asked 13 years, 4 months ago Modified 2 years ago Viewed 287k times

116

How to prevent default in an onclick method? I have a method in which I am also passing a custom value

```
<a href="#" onclick="callmymethod(24)">Call</a>
```

```
function callmymethod(myVal){
    //doing custom things with myVal
    //here I want to prevent default
}
```

javascript

Share

Improve this question

Follow

edited Jul 25, 2018 at 1:00

Ry- ♦

225k ● 56 ● 487 ● 495

asked Aug 14, 2011 at 11:58

coure2011

42.3k ● 86 ● 224 ● 359

13 Answers

Sorted by: Highest score (default)

143

Let your callback return `false` and pass that on to the `onclick` handler:

```
<a href="#" onclick="return callmymethod(24)">Call</a>

function callmymethod(myVal){
    //doing custom things with myVal
    //here I want to prevent default
    return false;
}
```

To create *maintainable* code, however, you should abstain from using *"inline Javascript"* (i.e.: code that's directly within an element's tag) and modify an element's behavior via an included Javascript source file (it's called unobtrusive Javascript).

The mark-up:

```
<a href="#" id="myAnchor">Call</a>
```

The code (separate file):

```
// Code example using Prototype JS API
$('myAnchor').observe('click', function(event) {
    Event.stop(event); // suppress default click behavior, cancel the event
    /* your onclick code goes here */
});
```

Share

Improve this answer

Follow

edited Sep 3, 2013 at 17:14

answered Aug 14, 2011 at 12:00

Linus Kleen

34.6k ● 10 ● 96 ● 100

1 @AamirAfridi Yes, it is working. You should adjust your fiddle and change `onLoaded` to `no wrap`. Or simply write your own .html markup instead. – Linus Kleen Jan 24, 2013 at 10:41

1 Please see answer below! "In my opinion the answer is wrong! He asked for event.preventDefault(); when you simply return false; it calls event.preventDefault(); AND event.stopPropagation(); as well!" – Ravindranath Akila Sep 3, 2013 at 13:05

1 To be fastidious, the OP's question never mentions `preventDefault`, the question is, how to *"prevent [the] default click"*. – Linus Kleen Sep 3, 2013 at 17:03

1 @LinusKleen, say also please, how to prevent click in this case – vp_arth Apr 11, 2014 at 7:53

1 @vp_arth Use "CSS click through" for this. Look at this question. – Linus Kleen Apr 11, 2014 at 17:56

71

In my opinion the answer is wrong! He asked for `event.preventDefault();` when you simply return false; it calls `event.preventDefault();` AND `event.stopPropagation();` as well!

You can solve it by this:

```
<a href="#" onclick="callmymethod(event, 24)">Call</a>
```

```
function callmymethod(e, myVal){
    //doing custom things with myVal

    //here I want to prevent default
    e = e || window.event;
    e.preventDefault();
}
```

Share

Improve this answer

Follow

edited Mar 6, 2019 at 1:43

Nigel Fds
813 ● 2 ● 16 ● 31

answered Mar 15, 2013 at 10:08

Wawa
719 ● 5 ● 2

6 You're relying on `window.event` which isn't always available across browsers.
– Linus Kleen Sep 3, 2013 at 17:04

Edited to add: event = event || window.event; – Nigel Fds Mar 6, 2019 at 0:54

Just as @LinusKleen said, the window.event is getting deprecated. developer.mozilla.org/en-US/docs/web/api/window/event – Hassan Faghihi Nov 1, 2021 at 7:14

3 Actually the original answer(without `e = e || window.event`) is NOT relying on `window.event`: developer.mozilla.org/en-US/docs/Web/HTML/… – YouJiacheng Oct 3, 2022 at 12:25

I agree, the accepted answer from Linus may work in some circumstances, but I couldn't get it to work using inline JS. My application dynamically creates and removes thousands of elements which need JS actions tied to them, so inline JS is really the best way of managing this without trying to manage thousands of event listeners dynamically. This method of passing the event to the function and then preventing propagation and defaults worked really well. – AutoBaker Jan 10 at 9:34

58

Try this (but please use buttons for such cases if you don't have a valid `href` value for graceful degradation)

```
<a href="#" onclick="callmymethod(24); return false;">Call</a>
```

Share

Improve this answer

Follow

edited Aug 4, 2020 at 10:21

answered Jan 24, 2013 at 9:54

Aamir Afridi
6,411 ● 3 ● 43 ● 42

1 nice simply best. I want same this for event.preventDefault(); – Sameer Kazi Apr 7, 2015 at 10:03

It worked for me, but could you explain why you have to `return false;`? What are we even returning false to? – Kellen Stuart Nov 9, 2018 at 20:22

you can just return instead of return false, returning will avoid link navigating to href. Technically we should use button for such cases. – Aamir Afridi Nov 12, 2018 at 9:35

16

Just place "javascript:void(0)", in place of "#" in href tag

```
<a href="javascript:void(0);" onclick="callmymethod(24)">Call</a>
```

Share Improve this answer Follow

answered Jul 1, 2014 at 8:54

Rafay
161 ● 1 ● 3

16

You can catch the event and then block it with preventDefault() -- works with pure Javascript

```
document.getElementById("xyz").addEventListener('click', function(event){
    event.preventDefault();
    console.log(this.getAttribute("href"));
    /* Do some other things*/
});
```

Share
Improve this answer
Follow

edited Feb 19, 2015 at 15:52

answered Feb 19, 2015 at 15:29

Andreas Karz
410 ● 5 ● 6

9

This worked for me

```
<a href="javascript:;" onclick="callmymethod(24); return false;">Call</a>
```

Share Improve this answer Follow

answered Apr 30, 2015 at 14:42

Pixelomo
6,737 ● 4 ● 48 ● 62

9

Another way to do that is to use the `event` object inside the attribute `onclick` (without the need to add an additional argument to the function to pass the event)

```
function callmymethod(myVal){
    console.log(myVal);
}
```

```
<a href="#link" onclick="event.preventDefault();callmymethod(24)">Call</a>
```

Run code snippet   Expand snippet

Share Improve this answer Follow

answered May 3, 2020 at 19:11

h3t1
1,226 ● 2 ● 19 ● 31

1 This should be the best solution – T. Phanelly Apr 11 at 8:53

7

You can use:

```
event.stopPropagation();
```

https://dom.spec.whatwg.org/#dom-event-stoppropagation

Share
Improve this answer
Follow

edited Jul 1, 2021 at 2:04
Koray Tugay
23.7k ● 49 ● 201 ● 332

answered Feb 15, 2018 at 11:06
Girdacio Pereira
123 ● 1 ● 3

6

Several different answers. Even after so many years, the question is still relevant. Therefore, here is the compilation:

```
// 1
function myFn1 (e, value) {
    console.log('value:', value)
    e.preventDefault();
}

// 2
function myFn2 (value) {
    console.log('value:', value)
    return false;
}

// 3,5,6,7
function myFn3 (value) {
    console.log('value:', value)
}

// 4
document.getElementById("clicki-1").addEventListener('click', function(event){
    event.preventDefault();
    console.log('value:',this.getAttribute("data-value"));
});
```

```
<h3>onclick Attribute</h3>
<div>
```

```
  <a href="/hello" onclick="myFn1(event, 42)">Click 1</a>
</div>

<div>
  <a href="/hello" onclick="return myFn2(42)">Click 2</a>
</div>

<div>
  <a href="/hello" onclick="myFn3(42); return false;">Click 3</a>
</div>

<h3>EventListener (addEventListener)</h3>
<div>
  <a href="/hello" id="clicki-1" data-value="42">Click 4</a>
</div>

<h3>onclick Attribute without linkaddress when hovering</h3>
<div>
  <a href="javascript:;" onclick="event.preventDefault(); myFn3(42)">Click
5</a>
</div>

<div>
  <a href="javascript:void(0);" onclick="myFn3(42)">Click 6</a>
</div>

<div>
  <a href="javascript:;" onclick="myFn3(42)">Click 7</a>
</div>

<h5>Set Anchor Hashtag</h5>
<div>
  <a href="#" onclick="myFn3(42)">Click 8</a>
</div>
```

Run code snippet   Expand snippet

I prefer the variant 2 with return false; if it has to go quickly. And otherwise variant 4 (AddEventListener).

Share Improve this answer Follow

answered Nov 22, 2021 at 22:57
Maik Lowrey
17.5k ● 7 ● 53 ● 98

1

It would be too tedious to alter function usages in all html pages to `return false`.

So here is a tested solution that patches only the function itself:

```
function callmymethod(myVal) {
    // doing custom things with myVal

    // cancel default event action
    var event = window.event || callmymethod.caller.arguments[0];
    event.preventDefault ? event.preventDefault() : (event.returnValue =
false);

    return false;
}
```

This correctly prevents IE6, IE11 and latest Chrome from visiting `href="#"` after onclick event handler completes.

Credits:

- How do you find out the caller function in JavaScript?
- event.preventDefault() function not working in IE

Share Improve this answer Follow

answered Apr 11, 2019 at 17:15
Vadzim
26.1k ● 14 ● 150 ● 159

0

If you need to put it in the tag. Not the finest solution, but it will work.

Make sure you put the onclick event in front of the href. Only worked for my this way.

```
<a onclick="return false;" href="//www.google.de">Google</a>
```

Share
Improve this answer
Follow

edited Jan 20, 2020 at 12:45
marc_s
753k ● 183 ● 1.4k ● 1.5k

answered Jan 14, 2020 at 16:09
Matix
128 ● 2 ● 11

0

Consider using a form instead of a link. Then, all you really need to do is add:

```
<input type="submit" onclick="event.preventDefault();">
```

You probably want to handle it though so in total you'd probably do something more like this:

```
function myFunction() {
  if (confirm("Are you sure you want to ...? This action cannot be undone."))
    document.getElementById("myForm").submit();
  }
}
```

```
<form method="post" action="/test" id="myForm">
  <input type="submit" onclick="event.preventDefault();myFunction();">
</form>
```

Run code snippet   Expand snippet

This allows the user to click ok to proceed or cancel to not have it submit the form.

Share
Improve this answer
Follow

edited Nov 26, 2022 at 4:18
Michael M.
11.1k ● 11 ● 21 ● 43

answered Nov 17, 2022 at 19:32
Brandon Wegner
66 ● 5

-3

Give a class or id to the element and use jquery function unbind();

```
$(".slide_prevent").click(function(){
                $(".slide_prevent").unbind();
              });
```

Share Improve this answer Follow

answered Aug 14, 2014 at 5:49
user46487
644 ● 6 ● 8