**Tricks**

1. **Cross-Browser Dependency-Free DOM Ready**

🡪The DOMContentLoaded event is fired when the initial HTML document has been completely loaded and parsed, without waiting for stylesheets, images, and subframes to finish loading.

🡪If you want the DOM to get parsed as fast as possible after the user has requested the page, you could make your JavaScript asynchronous and optimize loading of stylesheets.

**Note**: Synchronous JavaScript pauses parsing of the DOM.

***Snippet***

var DOMReady = function(a, b, c) {

b = document

c = 'addEventListener'

b[c] ? b[c]('DocumentContentLoaded', a) : window.attachEvent('onload', a)

}

DOMReady(function () {

alert('The DOM is Ready!');

});

The native DOM function is:

document.addEventListener('DOMContentLoaded', function() { });

1. **Format Currency**

🡪This function will round numbers to two decimal places, and ensure that the returned value has two decimal places. For example 12.006 will return 12.01, .3 will return 0.30, and 5 will return 5.00

***Snippet***

function currency(N){

N=parseFloat(N);

if(!isNaN(N))N=N.toFixed(2);

else N='0.00';return N;

}

Output:

currency("5"); - "5.00"

currency("5.009");-"5.01"

1. **Get All Possible DOM Events**

You can get an array of all the events that start with "on" (e.g. onclick) by running this in the Console of Firefox.

**Snippet**

[i for(i in document)].filter(function(i){

return i.substring(0,2)=='on'&&(document[i]==null||typeof document[i]=='function');

})

To get all Event listeners (in chrome)

**Snippet**

var items = Array.prototype.slice.call(

document.querySelectorAll('\*')

).map(function(element) {

var listeners = getEventListeners(element);

return {

element: element,

listeners: Object.keys(listeners).map(function(k) {

return { event: k, listeners: listeners[k] };

})

};

}).filter(function(item) {

return item.listeners.length;

});

1. **Get URL and URL Parts in JavaScript**

JavaScript can access the current URL in parts. For this URL:

*http://css-tricks.com/example/index.html?s=flexbox*

window.location.protocol = "http:"

window.location.host = "css-tricks.com"

window.location.pathname = "example/index.html"

window.location.search = "?s=flexbox"

So to get the full URL path in JavaScript:

**Snippet**

var newURL = window.location.protocol + "://" + window.location.host + "/" + window.location.pathname;

Output: "https:://secure.cbonline.co.uk//"

If you need to break up up the pathname, for example, a URL like http://css-tricks.com/blah/blah/blah/index.html, you can split the string on "/" characters

**Snippet 1 Snippet 2**

var pathArray =window.location.href.split('/');

var pathArray = window.location.pathname.split('/');

Output: Output:

(2) ["", ""] (4) ["https:", "", "secure.cbonline.co.uk", ""]

0: "" 0: "https:"

1: "" 1: ""

length: 2 2: "secure.cbonline.co.uk"

3: ""

length: 4

Then access the different parts by the parts of the array, like

var secondLevelLocation = pathArray[0];

Output (for snippet 1 pathArray)

var secondLevelLocation = pathArray[0];

undefined

secondLevelLocation

""

var secondLevelLocation = pathArray[1];

undefined

secondLevelLocation

""

Output (for snippet 2 pathArray)

var secondLevelLocation = pathArray[0];

undefined

secondLevelLocation

"https:"

var secondLevelLocation = pathArray[1];

undefined

secondLevelLocation

""

var secondLevelLocation = pathArray[2];

undefined

secondLevelLocation

"secure.cbonline.co.uk"

var secondLevelLocation = pathArray[3];

undefined

secondLevelLocation

""

To put that pathname back together, you can stitch together the array and put the "/"'s back in:

**Snippet**

var newPathname = "";

for (i = 0; i < pathArray.length; i++)

{

newPathname += "/";

newPathname += pathArray[i];

}

Output: (snippet 1) : "//"

Output (snippet 2) : "/https://secure.cbonline.co.uk/"

1. **Get URL Variables**

Example URL:

http://www.example.com/index.php?id=1&image=awesome.jpg

Calling getQueryVariable("id") - would return "1".

Calling getQueryVariable("image") - would return "awesome.jpg".

**Snippet**

function getQueryVariable(variable)

{

var query = window.location.search.substring(1);

var vars = query.split("&");

for (var i=0;i<vars.length;i++) {

var pair = vars[i].split("=");

if(pair[0] == variable){return pair[1];}

}

return(false);

}

URL: <https://apps.cybonline.co.uk/cyb/authenticate/cb?client_id=cb_web&response_type=token&state=aa274efe-a3ac-41ad-84ac-b7a9c2d5247b&redirect_uri=https://cybg.cbonline.co.uk/cyb/online-banking/bapp/authentication?applyFor=IM135&scope=30>

Output:

getQueryVariable("client\_id");

"cb\_web"

getQueryVariable("response\_type")

"token"

getQueryVariable("state")

"aa274efe-a3ac-41ad-84ac-b7a9c2d5247b"

getQueryVariable("redirect\_uri")

<https://cybg.cbonline.co.uk/cyb/online-banking/bapp/authentication?applyFor>

getQueryVariable("scope")

"30"

1. **Remove an Element**

For whatever reason, an element can't destroy itself in JavaScript. jQuery has a method for this, which is nice because this is how we think:

$(".remove-me").remove();

But there is no direct equivalent in JavaScript. Instead you'll need to select the parent element and use removeChild.

So if you have:

<div class="module">

<p>Stuff.</p>

<div class="remove-me">...</div>

</div>

You'll need to do:

**Snippet**

var thingToRemove = document.querySelectorAll(".remove-me")[0];

thingToRemove.parentNode.removeChild(thingToRemove);

Or if you had a reference to an element and wanted to empty out all the elements inside it, but keep it:

**Snippet**

mydiv = document.getElementById('empty-me');

while (mydiv.firstChild) {

mydiv.removeChild(mydiv.firstChild);

}

1. **Remove the Last Character from a String**

**Snippet**

var origString = 'Happy Dance7';

var trimmedString = origString.substring(0, origString.length-1);

console.log(trimmedString);

Output: var a=document.querySelector('span.hero--locator-font-hecto.hide-on-phones').innerText;

Undefined

a

“CASHBACK”

a.substring(0,a.length-2);

"CASHBA"

"Happy Dance7".slice(0,-1) //"Happy Dance"

**Snippet**

Output:

document.querySelector('span.hero--locator-font-hecto.hide-on-phones').innerText.slice(0,-1);

"CASHBAC"

document.querySelector('span.hero--locator-font-hecto.hide-on-phones').innerText.slice(0,1);

"C"

document.querySelector('span.hero--locator-font-hecto.hide-on-phones').innerText.slice(0,4);

"CASH"

document.querySelector('span.hero--locator-font-hecto.hide-on-phones').innerText.slice(0,-5);

"CAS"

document.querySelector('span.hero--locator-font-hecto.hide-on-phones').innerText.slice(0,5);

"CASHB"

1. **Toggle (Show/Hide) Element**

**Snippet**

function toggle\_visibility(id) {

var e = document.getElementById(id);

if(e.style.display == 'block')

e.style.display = 'none';

else

e.style.display = 'block';

}

Inline usage:

<a href="#" onclick="toggle\_visibility('foo');">Click here to toggle visibility of element #foo</a>

<div id="foo">This is foo</div>