

## Celestra cheatsheet – v5.6.4 – <https://github.com/Serrin/Celestra/>

The `celestra` and/or the `CEL` objects contain these functions, except the polyfills. Example: `CEL.qsa("p");`

Core API	DOM API	Type checking API
<code>javaHash(&lt;data&gt;[,hexa=false]);</code> <code>b64Encode(&lt;string&gt;); b64Decode(&lt;string&gt;);</code> <code>extend([deep,]&lt;target&gt;,&lt;source1&gt;[,srcN]);</code> <code>sizeIn(&lt;object&gt;);</code> <code>popIn(&lt;obj&gt;,&lt;prop&gt;);, forIn(&lt;obj&gt;,&lt;cb&gt;);</code> <code>filterIn(&lt;object&gt;,&lt;callback&gt;);</code> <code>delay + sleep(&lt;ms&gt;).then(&lt;callback&gt;);</code> <code>inherit(&lt;subclass&gt;,&lt;superclass&gt;);</code> <code>randomBoolean();</code> <code>randomUUIDv7();</code> <code>timestampID([size=21[,alphabet=BASE58]]);</code> <code>nanoid([size=21[,alphabet="A-Za-z0-9-_" ]]);</code> <code>BASE16; BASE32; BASE36; BASE58; BASE62;</code> <code>WORDSAFEALPHABET;</code> <code>getUrlVars([str=location.search]);</code> <code>obj2string(&lt;object&gt;);</code> <code>classof(&lt;variable&gt;[,type[,throw=false]]);</code> <code>bind(&lt;fn&gt;,&lt;context&gt;); and unBind(&lt;fn&gt;);</code> <code>constant(&lt;value&gt;); and identity(&lt;value&gt;);</code> <code>noop(); and T(); and F();</code> <code>assertEq(&lt;msg&gt;,&lt;v1&gt;,&lt;v2&gt;[,strict=true]);</code> <code>assertNotEq(&lt;m&gt;,&lt;v1&gt;,&lt;v2&gt;[,strict=true]);</code> <code>assertTrue(&lt;message&gt;,&lt;value&gt;);</code> <code>assertFalse(&lt;message&gt;,&lt;value&gt;);</code> <code>noConflict(); and VERSION;</code>	<code>qsa(&lt;selector&gt;[,context]).forEach(&lt;cb&gt;);</code> <code>qs(&lt;selector&gt;[,context]);</code> <code>domReady(&lt;callback&gt;);</code> <code>domCreate(&lt;type&gt;[,properties[,innerHTML]]);</code> <code>domCreate(&lt;element descriptive object&gt;);</code> <code>domToElement(&lt;htmlString&gt;);</code> <code>domGetCSS(&lt;element&gt;[,property]);</code> <code>domSetCSS(&lt;element&gt;,&lt;property&gt;,&lt;value&gt;);</code> <code>domSetCSS(&lt;element&gt;,&lt;properties&gt;);</code> <code>domFadeIn(&lt;element&gt;[,duration[,display]]);</code> <code>domFadeOut(&lt;element&gt;[,duration]);</code> <code>domFadeToggle(&lt;elem.&gt;[,duration[,display]]);</code> <code>domShow(&lt;element&gt;[,display]);</code> <code>domHide(&lt;element&gt;);</code> <code>domToggle(&lt;element&gt;[,display]);</code> <code>domIsHidden(&lt;element&gt;);</code> <code>domScrollToTop();</code> <code>domScrollToBottom();</code> <code>domScrollToElement(&lt;element&gt;[,top=true]);</code> <code>domSiblings(&lt;element&gt;);</code> <code>domSiblingsPrev(&lt;element&gt;);</code> <code>domSiblingsLeft(&lt;element&gt;);</code> <code>domSiblingsNext(&lt;element&gt;);</code> <code>domSiblingsRight(&lt;element&gt;);</code> <code>domGetCSSVar(&lt;name&gt;);</code> <code>domSetCSSVar(&lt;name&gt;,&lt;value&gt;);</code> <code>importScript(&lt;script1&gt;[,scriptN]);</code> <code>importStyle(&lt;style1&gt;[,styleN]);</code> <code>setFullscreenOn(&lt;selector&gt; or &lt;element&gt;);</code> <code>setFullscreenOff();</code> <code>getFullscreen();</code> <code>form2array(&lt;form&gt;);</code> <code>form2string(&lt;form&gt;);</code> <code>getDoNotTrack();</code> <code>getLocation(&lt;success&gt;[,error]);</code> <code>createFile(&lt;filename&gt;,&lt;content&gt;[,dType]);</code>	<code>isMap(&lt;value&gt;); and isWeakMap(&lt;v&gt;);</code> <code>isSet(&lt;value&gt;); and isWeakSet(&lt;v&gt;);</code> <code>isNumber(&lt;v&gt;); and isNumeric(&lt;v&gt;);</code> <code>isFloat(&lt;val&gt;); and isBigInt(&lt;v&gt;);</code> <code>isString(&lt;v&gt;); and isChar(&lt;val&gt;);</code> <code>isDate(&lt;val&gt;); and isError(&lt;val&gt;);</code> <code>isRegex(&lt;v&gt;); and isSymbol(&lt;v&gt;);</code> <code>isElement(&lt;v&gt;); and isObject(&lt;v&gt;);</code> <code>isDataView(&lt;value&gt;);</code> <code>isBoolean(&lt;value&gt;);</code> <code>isNull(&lt;value&gt;);</code> <code>isUndefined(&lt;value&gt;);</code> <code>isNullOrUndefined(&lt;v&gt;); isNil(&lt;v&gt;);</code> <code>isPlainObject(&lt;value&gt;);</code> <code>isTruthy(&lt;value&gt;); + isFalsy(&lt;v&gt;);</code> <code>isFunction(&lt;v&gt;); + isCallable(&lt;v&gt;);</code> <code>isConstructorFn(&lt;value&gt;);</code> <code>isGeneratorFn(&lt;value&gt;);</code> <code>isAsyncGeneratorFn(&lt;value&gt;);</code> <code>isAsyncFn(&lt;value&gt;);</code> <code>isArraylike(&lt;value&gt;);</code> <code>isTypedArray(&lt;value&gt;);</code> <code>isArrayBuffer(&lt;value&gt;);</code> <code>isPrimitive(&lt;value&gt;);</code> <code>isPromise(&lt;value&gt;);</code> <code>isIterator(&lt;v&gt;);</code> <code>isIterable(&lt;v&gt;);</code> <code>isEmptyObject(&lt;value&gt;);</code> <code>isEmptyArray(&lt;value&gt;);</code> <code>isEmptyMap(&lt;v&gt;); + isEmptySet(&lt;v&gt;);</code> <code>isEmptyIterator(&lt;value&gt;);</code> <code>isSameObject(&lt;object1&gt;,&lt;object2&gt;);</code> <code>isSameArray(&lt;array1&gt;,&lt;array2&gt;);</code> <code>isSameMap(&lt;map1&gt;,&lt;map2&gt;);</code> <code>isSameSet(&lt;set1&gt;,&lt;set2&gt;);</code> <code>isSameIterator(&lt;iter1&gt;,&lt;iter2&gt;);</code>
String API		
<code>strPropercase(&lt;str&gt;); strTitlecase(&lt;s&gt;);</code> <code>strCapitalize(&lt;string&gt;);</code> <code>strUpFirst(&lt;str&gt;); + strDownFirst(&lt;str&gt;);</code> <code>strReverse(&lt;s&gt;); + strCodePoints(&lt;s&gt;);</code> <code>strFromCodePoints(&lt;iterator&gt;);</code> <code>strAt(&lt;string&gt;,&lt;index&gt;[,newChar]);</code> <code>strSplice(&lt;str&gt;,&lt;index&gt;,&lt;count&gt;[,add]);</code> <code>strHTMLRemoveTags(&lt;string&gt;);</code> <code>strHTMLEscape(&lt;s&gt;); strHTMLUnEscape(&lt;s&gt;);</code>		

Collections API		Polyfills
<pre> arrayCreate([length=0]); arrayDeepClone(&lt;array&gt;); arrayMerge(&lt;target&gt;,&lt;source1&gt;[,sourceN]); arrayUnique(&lt;iterator&gt;); arrayAdd(&lt;array&gt;,&lt;value&gt;); arrayClear(&lt;array&gt;); arrayRemove(&lt;array&gt;,&lt;value&gt;[,all=false]); arrayRemoveBy(&lt;array&gt;,&lt;callback&gt;[,all=false]); arrayRange([start=0[,end=99[,step=1]]]); iterRange([start=0[,step=1[,end=Infinity]]]); arrayCycle(&lt;iterator&gt;[,n=100]); iterCycle(&lt;iter&gt;[,n=Infinity]); arrayRepeat(&lt;value&gt;[,n=100]); iterRepeat(&lt;value&gt;[,n=Infinity]); slice(&lt;iterator&gt;[,begin=0[,end=Infinity]]); without(&lt;iterator&gt;,&lt;filterIterator &gt;); reduce(&lt;iterator&gt;,&lt;callback&gt;[,initialvalue]); count(&lt;iterator&gt;,&lt;callback&gt;); take(&lt;iterator&gt;[,n=1]); takeWhile(&lt;iterator&gt;,&lt;callback&gt;); takeRight(&lt;iterator&gt;[,n=1]); takeRightWhile(&lt;iterator&gt;,&lt;callback&gt;); drop(&lt;iterator&gt;[,n=1]); dropWhile(&lt;iterator&gt;,&lt;callback&gt;); dropRight(&lt;iterator&gt;[,n=1]); dropRightWhile(&lt;iterator&gt;,&lt;callback&gt;); </pre>	<pre> forEach(&lt;iterator&gt;,&lt;callback&gt;); map(&lt;iterator&gt;,&lt;callback&gt;); enumerate(&lt;iterator&gt;[,offset=0]); entries(&lt;iterator&gt;[,offset=0]); size(&lt;iterator&gt;); every(&lt;iterator&gt;,&lt;callback&gt;); some(&lt;iterator&gt;,&lt;callback&gt;); none(&lt;iterator&gt;,&lt;callback&gt;); includes(&lt;iterator&gt;,&lt;value&gt;); contains(&lt;iterator&gt;,&lt;value&gt;); find(&lt;iterator&gt;,&lt;callback&gt;); findLast(&lt;iterator&gt;,&lt;callback&gt;); filter(&lt;iterator&gt;,&lt;callback&gt;); reject(&lt;iterator&gt;,&lt;callback&gt;); partition(&lt;iterator&gt;,&lt;callback&gt;); zip(&lt;iterator1&gt;[,iteratorN]); unzip(&lt;iterator&gt;); zipObj(&lt;iterator1&gt;,&lt;iterator2&gt;); shuffle(&lt;iterator&gt;); min(&lt;value1&gt;[,valueN]); max(&lt;value1&gt;[,valueN]); sort(&lt;iterator&gt;[,numbers=false]); reverse(&lt;iterator&gt;); item(&lt;iterator&gt;,&lt;index&gt;); nth(&lt;iterator&gt;,&lt;index&gt;); first(&lt;iterator&gt;); head(&lt;iterator&gt;); last(&lt;iterator&gt;); initial(&lt;iterator&gt;); tail(&lt;iterator&gt;); flat(&lt;iterator&gt;); concat(&lt;iterator1&gt;[,iteratorN]); join(&lt;iterator&gt;[,separator=","]); </pre>	<pre> Array.fromAsync(); Array.prototype.toReversed(); Array.prototype.toSorted(); Array.prototype.toSpliced(); Array.prototype.with(); crypto.randomUUID(); Error.isError(); globalThis; Map.groupBy(); Object.groupBy(); Object.hasOwn(); TypedArray.prototype.toReversed(); TypedArray.prototype.toSorted(); TypedArray.prototype.with(); </pre>
		Non-standard polyfills
		<pre> BigInt.prototype.toJSON(); window.AsyncFunction(); window.GeneratorFunction(); </pre>

Math API		Abstract API
<code>sum(&lt;value1&gt;[,valueN]);</code> <code>avg(&lt;value1&gt;[,valueN]);</code> <code>product(&lt;value1&gt;[,valueN]);</code>  <code>clamp(&lt;value&gt;,&lt;min&gt;,&lt;max&gt;);</code> <code>minmax(&lt;value&gt;,&lt;min&gt;,&lt;max&gt;);</code>  <code>isEven(&lt;value&gt;);</code> <code>isOdd(&lt;value&gt;);</code>  <code>randomInt([max]);</code> <code>randomInt(&lt;min&gt;,&lt;max&gt;);</code>  <code>randomFloat([max]);</code> <code>randomFloat(&lt;min&gt;,&lt;max&gt;);</code>  <code>inRange(&lt;value&gt;,&lt;min&gt;,&lt;max&gt;);</code> <code>signbit(&lt;value&gt;);</code>	<code>toInt8(&lt;value&gt;);</code> <code>toInt16(&lt;value&gt;);</code> <code>toInt32(&lt;value&gt;);</code> <code>toUInt8(&lt;value&gt;);</code> <code>toUInt16(&lt;value&gt;);</code> <code>toUInt32(&lt;value&gt;);</code> <code>toBigInt64(&lt;value&gt;);</code> <code>toBigUint64(&lt;value&gt;);</code> <code>toFloat16(&lt;value&gt;);</code> <code>toFloat32(&lt;value&gt;);</code>  <code>isInt8(&lt;value&gt;);</code> <code>isInt16(&lt;value&gt;);</code> <code>isInt32(&lt;value&gt;);</code> <code>isUInt8(&lt;value&gt;);</code> <code>isUInt16(&lt;value&gt;);</code> <code>isUInt32(&lt;value&gt;);</code> <code>isBigInt64(&lt;value&gt;);</code> <code>isBigUint64(&lt;value&gt;);</code> <code>isFloat16(&lt;value&gt;);</code>	<code>getIn(&lt;object&gt;,&lt;property&gt;);</code> <code>getInV(&lt;object&gt;,&lt;property&gt;);</code> <code>hasIn(&lt;object&gt;,&lt;property&gt;);</code> <code>setIn(&lt;object&gt;,&lt;property&gt;,&lt;value&gt;);</code>  <code>toIndex(&lt;value&gt;);</code> <code>toPropertyKey(&lt;value&gt;);</code> <code>toInteger(&lt;value&gt;);</code> <code>toArray(value);</code> <code>toObject(&lt;value&gt;);</code>  <code>isIndex(&lt;value&gt;);</code> <code>isPropertyKey(&lt;value&gt;);</code> <code>isSameValue(&lt;value1&gt;,&lt;value2&gt;);</code> <code>isSameValueZero(&lt;value1&gt;,&lt;value2&gt;);</code> <code>isSameValueNonNumber(&lt;value1&gt;,&lt;value2&gt;);</code>  <code>type(&lt;value&gt;);</code> <code>createDataProperty(&lt;object&gt;,&lt;property&gt;,&lt;value&gt;);</code> <code>createMethodProperty(&lt;object&gt;,&lt;property&gt;,&lt;value&gt;);</code>
Cookie API		
<code>getCookie([name]);</code> <code>hasCookie(&lt;name&gt;);</code> <code>setCookie(&lt;Options object&gt;);</code> <code>setCookie(&lt;name&gt;,&lt;value&gt;[,hours=8760[,path="/"[,domain[,secure[,SameSite="Lax"[,HttpOnly]]]]]]];</code> <code>removeCookie(&lt;Options object&gt;);</code> <code>removeCookie(&lt;name&gt;[,path="/"[,domain[,secure[,SameSite="Lax"[,HttpOnly]]]]];</code> <code>clearCookies(&lt;Options object&gt;);</code> <code>clearCookies([path="/"[,domain[,sec[,SameSite="Lax"[,HttpOnly]]]]];</code>		
AJAX and CORS API		
<code>getText(&lt;url&gt;,&lt;success&gt;);</code> <code>getJSON(&lt;url&gt;,&lt;success&gt;);</code>  <code>ajax(&lt;Options object&gt;);</code> <b>Options object properties (* = default value):</b> <code>url: string, data: string, queryType: "ajax"/"cors", type: "get"/"post", success: function, error: function, format: "text"/"json"/"xml", user: string, password: string</code>		

Removed Polyfills - Available in celestra-polyfills.dev.js and celestra-polyfills.min.js		
v3.1.0	v3.8.0	v5.6.0
Array.from(); Array.of(); Array.prototype.copyWithin(); Array.prototype.fill(); Array.prototype.find(); Array.prototype.findIndex(); Object.create(); String.fromCodePoint(); String.prototype.codePointAt(); String.prototype.endsWith(); String.prototype.startsWith(); Math.acosh(); Math.asinh(); Math.atanh(); Math.cbrt(); Math.clz32(); Math.cosh(); Math.expml(); Math.fround(); Math.hypot(); Math.imul(); Math.log1p(); Math.log10(); Math.log2(); Math.sign(); Math.sinh(); Math.tanh(); Math.trunc(); Number.EPSILON; Number.isNaN(); and isNaN(); Number.isInteger(); Number.isFinite(); Number.isSafeInteger(); Number.parseInt(); Number.parseFloat();	Array.prototype.values(); Array.prototype.includes(); ChildNode.after(); ChildNode.before(); ChildNode.remove(); ChildNode.replaceWith(); Element.prototype.closest(); Element.prototype.getAttributeNames(); Element.prototype.matches(); Element.prototype.toggleAttribute(); ParentNode.append(); ParentNode.prepend(); String.prototype[Symbol.iterator](); String.prototype.includes(); String.prototype.repeat(); NodeList.prototype.forEach(); Object.assign(); Object.entries(); Object.getOwnPropertyDescriptors(); Object.values(); RegExp.prototype.flags; window.screenLeft; window.screenTop;	Array.prototype.at(); Array.prototype.findLast(); Array.prototype.findLastIndex(); Array.prototype.flat(); Array.prototype.flatMap(); Array.prototype.group(); Array.prototype.groupToMap(); Number.MIN_SAFE_INTEGER; Number.MAX_SAFE_INTEGER; Object.fromEntries(); Object.is(); String.prototype.at(); String.prototype.matchAll(); String.prototype.padStart(); String.prototype.padEnd(); String.prototype.replaceAll(); String.prototype.trimStart(); String.prototype.trimLeft(); String.prototype.trimEnd(); String.prototype.trimRight(); Typedarray.prototype.at(); TypedArray.prototype.findLast(); TypedArray.prototype.findLastIndex();