

## Celestra cheatsheet – v4.4.2 – <https://github.com/Serrin/Celestra/>

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

Core API	DOM	Type checking
<code>signbit(&lt;value&gt;;</code> <code>delay(&lt;ms&gt;).then(&lt;callback&gt;;</code> <code>inherit(&lt;subclass&gt;,&lt;superclass&gt;;</code> <code>randomInt([&lt;max&gt; or &lt;min&gt;,&lt;max&gt;;</code> <code>randomFloat([&lt;max&gt; or &lt;min&gt;,&lt;max&gt;;</code> <code>randomBoolean();</code> <code>randomString([&lt;length&gt;,&lt;specChar&gt;;</code> <code>b64Encode(&lt;string&gt;;, b64Decode(&lt;str&gt;;</code> <code>javaHash(&lt;data&gt;[,&lt;hexa&gt;;</code> <code>getUrlVars([&lt;str&gt;=&lt;location.search&gt;;</code> <code>obj2string(&lt;object&gt;;</code> <code>getType(&lt;variable&gt;[,&lt;type&gt;;</code> <code>extend([&lt;deep&gt;,&lt;target&gt;,&lt;source1&gt;[,&lt;srcN&gt;;</code> <code>deepAssign(&lt;target&gt;,&lt;source1&gt;[,&lt;srcN&gt;;</code> <code>sizeIn(&lt;obj&gt;; and forIn(&lt;obj&gt;,&lt;cb&gt;;</code> <code>filterIn(&lt;object&gt;,&lt;callback&gt;;</code> <code>popIn(&lt;object&gt;,&lt;property&gt;;</code> <code>strCapitalize(&lt;string&gt;;</code> <code>strUpFirst(&lt;str&gt;;, strDownFirst(&lt;str&gt;;</code> <code>strHTMLRemoveTags(&lt;string&gt;;</code> <code>strHTMLEscape(&lt;string&gt;;</code> <code>strHTMLUnEscape(&lt;string&gt;;</code> <code>strReverse(&lt;string&gt;;</code> <code>strCodePoints(&lt;string&gt;;</code> <code>strFromCodePoints(&lt;collection&gt;;</code> <code>strAt(&lt;string&gt;,&lt;index&gt;;</code> <code>toFunction(&lt;fn&gt;;, bind(&lt;fn&gt;,&lt;context&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;[,&lt;strict&gt;=true];</code> <code>assertNotEq(&lt;m&gt;,&lt;v1&gt;,&lt;v2&gt;[,&lt;strict&gt;=true];</code> <code>assertTrue(&lt;msg&gt;,&lt;value&gt;;</code> <code>assertFalse(&lt;msg&gt;,&lt;value&gt;;</code> <code>noConflict(); and VERSION;</code>	<code>qsa(&lt;selector&gt;[,&lt;context&gt;]).forEach(&lt;cb&gt;;</code> <code>qs(&lt;selector&gt;[,&lt;context&gt;;</code> <code>domReady(&lt;callback&gt;;</code> <code>domCreate(&lt;type&gt;[,&lt;properties&gt;[,&lt;innerHTML&gt;];</code> <code>domCreate(&lt;element descriptive object&gt;;</code> <code>domToElement(&lt;htmlString&gt;;</code> <code>domGetCSS(&lt;element&gt;[,&lt;property&gt;;</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;[,&lt;duration&gt;[,&lt;display&gt;];</code> <code>domFadeOut(&lt;element&gt;[,&lt;duration&gt;;</code> <code>domFadeToggle(&lt;elem.&gt;[,&lt;duration&gt;[,&lt;display&gt;];</code> <code>domShow(&lt;element&gt;[,&lt;display&gt;;</code> <code>domHide(&lt;element&gt;;</code> <code>domToggle(&lt;element&gt;[,&lt;display&gt;;</code> <code>domIsHidden(&lt;element&gt;;</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;url&gt;[,&lt;success&gt;;</code> <code>importScripts(&lt;scripts&gt; or &lt;script1&gt;[,&lt;scN&gt;;</code> <code>importStyle(&lt;href&gt;[,&lt;success&gt;;</code> <code>importStyles(&lt;styles&gt; or &lt;style1&gt;[,&lt;styleN&gt;;</code> <code>setFullscreenOn(&lt;selector&gt; or &lt;element&gt;;</code> <code>setFullscreenOff();</code> <code>getFullscreen();</code> <code>form2array(&lt;form&gt;; and form2string(&lt;form&gt;;</code> <code>getDoNotTrack();</code> <code>getLocation(&lt;success&gt;[,&lt;error&gt;;</code> <code>createFile(&lt;filename&gt;,&lt;content&gt;[,&lt;dType&gt;;</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;v&gt;; and isBigInt(&lt;v&gt;;</code> <code>isString(&lt;v&gt;; and isChar(&lt;v&gt;;</code> <code>isDate(&lt;v&gt;; and isError(&lt;v&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>isNull(&lt;value&gt;;</code> <code>isUndefined(&lt;value&gt;;</code> <code>isNullOrUndefined(&lt;value&gt;;</code> <code>isNil(&lt;value&gt;;</code> <code>isFunction(&lt;value&gt;;</code> <code>isGeneratorFn(&lt;value&gt;;</code> <code>isAsyncFn(&lt;value&gt;;</code> <code>isDataView(&lt;value&gt;;</code> <code>isBoolean(&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>isIterator(&lt;value&gt;;</code> <code>isIterable(&lt;value&gt;;</code> <code>isPromise(&lt;value&gt;;</code> <code>isEmptyObject(&lt;value&gt;;</code> <code>isEmptyArray(&lt;value&gt;;</code> <code>isEmptyMap(&lt;value&gt;;</code> <code>isEmptySet(&lt;value&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;set&gt;,&lt;set2&gt;;</code> <code>isSameIterator(&lt;iter1&gt;,&lt;iter2&gt;;</code>
AJAX and CORS		
<code>ajax(&lt;Options object&gt;;, getJson(&lt;url&gt;,&lt;success&gt;;, getText(&lt;url&gt;,&lt;success&gt;;</code> <b>Options object properties (* = default value):</b> url: <i>string</i> , data: <i>string</i> , queryType: <i>*"ajax"/"cors"</i> , type: <i>*"get"/"post"</i> , success: <i>function</i> , error: <i>function</i> , format: <i>*"text"/"json"/"xml"</i> , user: <i>string</i> , password: <i>string</i>		

Collections		Polyfills
<pre> arrayMerge([deep, ]&lt;target&gt;, &lt;source1&gt;[, srcN]); arrayUnique(&lt;collection&gt;); arrayAdd(&lt;array&gt;, &lt;value&gt;); arrayClear(&lt;array&gt;); arrayRemove(&lt;array&gt;, &lt;value&gt;[, all]); arrayRange([start=0[, end=100[, step=1]]]); arrayCycle(&lt;collection&gt;[, n]); arrayRepeat(&lt;value&gt;[, n]); iterRange([start=0[, step=1[, end]]]); iterCycle(&lt;iter&gt;[, n]); iterRepeat(&lt;value&gt;[, n]);  arrayUnion(&lt;collection1&gt;[, collectionN]); arrayIntersection(&lt;collection1&gt;, &lt;collection2&gt;); arrayDifference(&lt;collection1&gt;, &lt;collection2&gt;); arraySymmetricDifference(&lt;collec1&gt;, &lt;collec2&gt;); setUnion(&lt;collection1&gt;[, collectionN]); setIntersection(&lt;set1&gt;, &lt;set2&gt;); setDifference(&lt;set1&gt;, &lt;set2&gt;); setSymmetricDifference(&lt;set1&gt;, &lt;set2&gt;); isSuperset(&lt;superCollection&gt;, &lt;subCollection&gt;);  without(&lt;collection&gt;, &lt;filterCollection&gt;); reduce(&lt;collection&gt;, &lt;callback&gt;[, initialValue]);  take(&lt;collection&gt;[, n]); takeWhile(&lt;collection&gt;, &lt;callback&gt;); takeRight(&lt;collection&gt;[, n]); takeRightWhile(&lt;collection&gt;, &lt;callback&gt;); drop(&lt;collection&gt;[, n]); dropWhile(&lt;collection&gt;, &lt;callback&gt;); dropRight(&lt;collection&gt;[, n]); dropRightWhile(&lt;collection&gt;, &lt;callback&gt;); </pre>	<pre> forEach(&lt;collect.&gt;, &lt;callback&gt;); map(&lt;collection&gt;, &lt;callback&gt;); enumerate(&lt;collection&gt;[, offset=0]); entries(&lt;collection&gt;[, offset=0]); size(&lt;collection&gt;); every(&lt;collection&gt;, &lt;callback&gt;); some(&lt;collection&gt;, &lt;callback&gt;); none(&lt;collection&gt;, &lt;callback&gt;); includes(&lt;collection&gt;, &lt;value&gt;); find(&lt;collection&gt;, &lt;callback&gt;); filter(&lt;collection&gt;, &lt;callback&gt;); reject(&lt;collection&gt;, &lt;callback&gt;); partition(&lt;collection&gt;, &lt;callback&gt;); groupBy(&lt;collection&gt;, &lt;callback&gt;); min(&lt;collection&gt;); max(&lt;collection&gt;); sort(&lt;collection&gt;[, numberSort]); reverse(&lt;collection&gt;); shuffle(&lt;collection&gt;); zip(&lt;collection1&gt;[, collectionN]); unzip(&lt;collection&gt;);  item(&lt;collection&gt;, &lt;index&gt;); nth(&lt;collection&gt;, &lt;index&gt;); first(&lt;collection&gt;); head(&lt;collection&gt;); last(&lt;collection&gt;); initial(&lt;collection&gt;); tail(&lt;collection&gt;); slice(&lt;collection&gt;[, begin[, end]]); flat(&lt;collection&gt;); concat(&lt;collection1&gt;[, collectionN]); join(&lt;collection&gt;[, separator=","]); </pre>	<pre> Array.prototype.at(&lt;index&gt;); Array.prototype.findLast(); Array.prototype.findLastIndex(); Array.prototype.flat(); Array.prototype.flatMap();  globalThis;  Object.fromEntries(); Object.hasOwn();  String.prototype.at(&lt;index&gt;); 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(&lt;index&gt;); TypedArray.prototype.findLast(); TypedArray.prototype.findLastIndex(); </pre>
		Non-standard polyfills
		<pre> BigInt.prototype.toJSON(); window.AsyncFunction(); window.GeneratorFunction(); </pre>
Cookie		
<pre> getCookie([name]);, hasCookie(&lt;name&gt;);, setCookie(&lt;name&gt;, &lt;value&gt;[, hours=8760[, path="/"[, domain[, secure[, SameSite="Lax"[, HttpOnly]]]]]]);, setCookie(&lt;Optionsobj&gt;); removeCookie(&lt;name&gt;[, path="/"[, domain[, secure[, SameSite="Lax"[, HttpOnly]]]]]);, removeCookie(&lt;Options object&gt;);, clearCookies([path="/"[, domain[, sec[, SameSite="Lax"[, HttpOnly]]]]]);, clearCookies(&lt;Options object&gt;); </pre>		