

Celestra cheatsheet – v3.1.1 – <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>inherit(<subclass>,<superclass>);</code> <code>random(<min>,<max> or [max]);</code> <code>randomString([length[,specChar]]);</code> <code>b64Encode(<string>);</code> <code>b64Decode(<string>);</code> <code>javaHash(<data>[,hexa]);</code> <code>getUrlVar([name]);</code> <code>getUrlVarFromString(<querystr>[,name]);</code> <code>obj2string(<object>);</code> <code>getType(<variable>[,type]);</code> <code>extend([deep,]<target>,<source1>[,srcN]);</code> <code>deepAssign(<target>,<source1>[,srcN]);</code> <code>forIn(<object>,<callback>);</code> <code>mapIn(<object>,<callback>);</code> <code>strRemoveTags(<string>);</code> <code>strReverse(<string>);</code> <code>strReplaceAll(<str>,<search>,<replace>);</code> <code>toFunction(<function>);</code> <code>bind(<function>,<context>);</code> <code>hasOwn(<object>,<property>);</code> <code>constant(<value>);</code> <code>identity(<value>);</code> <code>noop();</code> <code>T();</code> <code>F();</code> <code>noConflict();</code> <code>VERSION;</code>	<code>qsa(<selector>[,context]).forEach(<fn>);</code> <code>qs(<selector>[,context]).argument;</code> <code>domReady(<function>);</code> <code>domCreate(<type>[,properties[,innerHTML]]);</code> <code>domCreate(<element descriptive object>);</code> <code>domToElement(<htmlString>);</code> <code>domGetCSS(<element>,<property>);</code> <code>domSetCSS(<element>,<property>,<value>);</code> <code>domSetCSS(<element>,<properties>);</code> <code>domFadeIn(<element>[,duration[,display]]);</code> <code>domFadeOut(<element>[,duration]);</code> <code>domFadeToggle(<elem.>[,duration[,display]]);</code> <code>domShow(<element>[,display]);, domHide(<el>);</code> <code>domToggle(<element>[,display]);</code> <code>domIsHidden(<element>);</code> <code>domOn(<eventTarget>,<eventType>,<callback>);</code> <code>domOff(<eventTarget>,<eventType>,<callback>);</code> <code>domTrigger(<eventTarget>,<eventType>);</code> <code>domSiblings(<element>);</code> <code>importScript(<url>[,success]);</code> <code>importScripts(<scripts> or <script1>[,scN]);</code> <code>importStyle(<href>[,success]);</code> <code>importStyles(<styles> or <style1>[,styleN]);</code> <code>setFullscreenOn(<selector> or <element>);</code> <code>setFullscreenOff();</code> <code>getFullscreen();</code> <code>form2array(<form>); and form2string(<form>);</code> <code>getDoNotTrack();</code> <code>getLocation(<success>[,error]);</code> <code>createFile(<filename>,<content>[,dType]);</code>	<code>isEqual(<value1>,<value2>); ES5</code> <code>isString(<v>); and isChar(<v>);</code> <code>isNumber(<value>);</code> <code>isNumeric(<value>);</code> <code>isFloat(<value>);</code> <code>isBigInt(<value>);</code> <code>isDate(<value>);</code> <code>isBoolean(<value>);</code> <code>isElement(<value>);</code> <code>isObject(<value>);</code> <code>isEmptyObject(<value>);</code> <code>isFunction(<value>);</code> <code>isArraylike(<value>);</code> <code>isEmptyArray(<value>);</code> <code>isTypedArray(<value>);</code> <code>isArrayBuffer(<value>);</code> <code>isNull(<value>);</code> <code>isUndefined(<value>);</code> <code>isNullOrUndefined(<value>);</code> <code>isNil(<value>);</code> <code>isPrimitive(<value>);</code> <code>isRegexp(<value>);</code> <code>isSymbol(<value>);</code> <code>isMap(<value>);</code> <code>isWeakMap(<value>);</code> <code>isSet(<value>);</code> <code>isWeakSet(<value>);</code> <code>isIterator(<value>);</code> <code>isIterable(<value>);</code> <code>isGenerator(<value>);</code>
AJAX and CORS		
<code>ajax(<Options obj.>);, getJson(<url>,<success>);, getText(<url>,<success>);</code>		
Options object properties (* = default value): url: string, data: string, queryType: <i>"ajax"/"cors"</i> , type: <i>"get"/"post"</i> , success: function, error: function, format: <i>"text"/"json"/"xml"</i> , user: string, password: string		
Cookie		
<code>setCookie(<name>,<value>[,hours[,path[,domain[,secure[,HttpOnly]]]]];, getCookie([name]);, hasCookie(<name>);,</code> <code>removeCookie(<name>[,path[,domain[,secure[,HttpOnly]]]]];, clearCookies([path[,domain[,secure[,HttpOnly]]]]];</code>		

Collections	Polyfills
<pre> isSuperset(<superset>,<subset>); arrayMerge([deep,]<target>,<source1>[,srcN]); zip(<collection1>[,collectionN]); and unzip(<collection>); uniqueArray(<value>); and uniquePush(<array>,<value>); arrayClear(<array>); and arrayRemove(<array>,<value>[,all]); min(<collection>); and minIndex(<collection>); max(<collection>); and maxIndex(<collection>); setUnion(<collection1>[,collectionN]); setIntersection(<set1>,<set2>); setDifference(<set1>,<set2>); setSymmetricDifference(<set1>,<set2>); arrayUnion(<collection1>[,collectionN]); arrayIntersection(<collection1>,<collection2>); arrayDifference(<collection1>,<collection2>); arraySymmetricDifference(<collection1>,<collection2>); arrayRange(<start>,<end>[,step]); and iterRange([start[,step[,end]]]); arrayCycle(<collection>[,n]); and iterCycle(<iter>[,n]); arrayRepeat(<value>[,n]);, and iterRepeat(<value>[,n]); sizeof(<collection>); item(<collection>,<index>); and itemOf(<collection>,<index>); forEach(<collection>,<callback>); and forOf(<collection>,<callback>); map(<collection>,<callback>); and mapOf(<collection>,<callback>); filterOf(<collection>,<callback>); hasOf(<collection>,<value>); findOf(<collection>,<callback>); everyOf(<collection>,<callback>); someOf(<collection>,<callback>); noneOf(<collection>,<callback>); firstOf(<collection>); and lastOf(<collection>); sliceOf(<collection>[,begin[,end]]); reverseOf(<collection>); sortOf(<collection>); reduceOf(<collection>,<callback>[,initialvalue]); concatOf(<collection1>[,collectionN]); enumerateOf(<collection>); takeOf(<collection>[,n]); and takeWhile(<collection>,<callback>); takeRight(<collection>[,n]); and takeRightWhile(<collection>,<callback>); dropOf(<collection>[,n]); and dropWhile(<collection>,<callback>); dropRight(<collection>[,n]); and dropRightWhile(<collection>,<callback>); </pre>	<pre> Array.prototype.values(); Array.prototype.includes(); Array.prototype.flat(); Array.prototype.flatMap(); String.prototype.includes(); String.prototype.trimStart(); String.prototype.trimLeft(); String.prototype.trimEnd(); String.prototype.trimRight(); String.prototype.padStart(); String.prototype.padEnd(); String.prototype.repeat(); String.prototype.matchAll(); String.prototype[Symbol.iterator](); Object.assign(); Object.fromEntries(); Object.entries(); Object.values(); Object.getOwnPropertyDescriptors(); RegExp.prototype.flags; NodeList.prototype.forEach(); ChildNode.after(); ChildNode.before(); ChildNode.remove(); ChildNode.replaceWith(); ParentNode.append(); ParentNode.prepend(); Element.prototype.matches(); Element.prototype.closest(); Element.prototype.toggleAttribute(); Element.prototype.getAttributeNames(); window.screenLeft; window.screenTop; globalThis; GeneratorFunction(); </pre>