

Celestra cheatsheet – v5.4.1 – <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	Type checking
<code>signbit(<value>);</code> <code>delay(<ms>).then(<callback>);</code> <code>inherit(<subclass>,<superclass>);</code> <code>randomInt([<max> or <min>,<max>);</code> <code>randomFloat([<max> or <min>,<max>);</code> <code>randomBoolean();</code> <code>randomID([hyphens=true],[usedate=false]);</code> <code>randomString([length[,specChar=false]]);</code> <code>inRange(<value>,<min>,<max>);</code> <code>b64Encode(<string>);, b64Decode(<str>);</code> <code>javaHash(<data>[,hexa=false]);</code> <code>getUrlVars([str=location.search]);</code> <code>obj2string(<object>);</code> <code>classof(<variable>[,type[,throw=false]]);</code> <code>extend([<deep>,<target>,<source1>[,<srcN>]);</code> <code>sizeIn(<obj>); and forIn(<obj>,<cb>);</code> <code>filterIn(<obj>,<cb>);, popIn(<obj>,<pr>);</code> <code>strPropercase(<s>); + strTitlecase(<s>);</code> <code>strCapitalize(<s>);</code> <code>strUpFirst(<str>);, strDownFirst(<str>);</code> <code>strHTMLRemoveTags(<string>);</code> <code>strHTMLEscape(<s>);, strHTMLUnEscape(<s>);</code> <code>strReverse(<str>);, strAt(<str>,<index>);</code> <code>strCodePoints(<string>);</code> <code>strFromCodePoints(<collection>);</code> <code>bind(<fn>,<context>); and unBind(<fn>);</code> <code>constant(<value>); and identity(<value>);</code> <code>noop(); and T(); and F();</code> <code>assertEq(<msg>,<v1>,<v2>[,strict=true]);</code> <code>assertNotEq(<m>,<v1>,<v2>[,strict=true]);</code> <code>assertTrue(<msg>,<value>);</code> <code>assertFalse(<msg>,<value>);</code> <code>noConflict(); and VERSION;</code>	<code>qsa(<selector>[,context]).forEach(<cb>);</code> <code>qs(<selector>[,context]);</code> <code>domReady(<callback>);</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]);</code> <code>domHide(<element>);</code> <code>domToggle(<element>[,display]);</code> <code>domIsHidden(<element>);</code> <code>domSiblings(<element>);</code> <code>domSiblingsPrev(<element>);</code> <code>domSiblingsLeft(<element>);</code> <code>domSiblingsNext(<element>);</code> <code>domSiblingsRight(<element>);</code> <code>domGetCSSVar(<name>);</code> <code>domSetCSSVar(<name>,<value>);</code> <code>importScript(<script1>[,scriptN]);</code> <code>importStyle(<style1>[,styleN]);</code> <code>setFullscreenOn(<selector> or <element>);</code> <code>setFullscreenOff();</code> <code>getFullscreen();</code> <code>form2array(<form>);</code> <code>form2string(<form>);</code> <code>getDoNotTrack();</code> <code>getLocation(<success>[,error]);</code> <code>createFile(<filename>,<content>[,dType]);</code>	<code>isMap(<value>); and isWeakMap(<v>);</code> <code>isSet(<value>); and isWeakSet(<v>);</code> <code>isNumber(<v>); and isNumeric(<v>);</code> <code>isFloat(<val>); and isBigInt(<v>);</code> <code>isString(<v>); and isChar(<val>);</code> <code>isDate(<val>); and isError(<val>);</code> <code>isRegex(<v>); and isSymbol(<v>);</code> <code>isElement(<v>); and isObject(<v>);</code> <code>isDataView(<val>);, isBoolean(<v>);</code> <code>isNull(<val>);, isUndefined(<val>);</code> <code>isNullOrUndefined(<v>); isNil(<v>);</code> <code>isPlainObject(<value>);</code> <code>isTruthy(<value>);, isFalsy(<val>);</code> <code>isFunction(<v>); + isCallable(<v>);</code> <code>isConstructorFn(<value>);</code> <code>isGeneratorFn(<value>);</code> <code>isAsyncGeneratorFn(<value>);</code> <code>isAsyncFn(<value>);</code> <code>isArraylike(<value>);</code> <code>isTypedArray(<value>);</code> <code>isArrayBuffer(<value>);</code> <code>isPrimitive(<v>);, isPromise(<v>);</code> <code>isIterator(<v>);, isIterable(<v>);</code> <code>isEmptyObject(<value>);</code> <code>isEmptyArray(<value>);</code> <code>isEmptyMap(<value>);</code> <code>isEmptySet(<value>);</code> <code>isEmptyIterator(<value>);</code> <code>isSameObject(<object1>,<object2>);</code> <code>isSameArray(<array1>,<array2>);</code> <code>isSameMap(<map1>,<map2>);</code> <code>isSameSet(<set1>,<set2>);</code> <code>isSameIterator(<iter1>,<iter2>);</code>
AJAX and CORS		
<code>ajax(<Options object>);, getJson(<url>,<success>);, getText(<url>,<success>);</code> Options object properties (* = default value): <code>url: string, data: string, queryType: *"ajax"/"cors", type: *"get"/"post", success: function, error: function, format: *"text"/"json"/"xml", user: string, password: string</code>		

Collections		Polyfills
arrayCreate ([length=0]); arrayDeepClone (<array>); arrayMerge (<target>,<source1>[, sourceN]); arrayUnique (<collection>); arrayAdd (<array>,<value>); arrayClear (<array>); arrayRemove (<array>,<value>[,all=false]); arrayRemoveBy (<array>,<callback>[,all=false]); arrayRange ([start=0[,end=99[,step=1]]]); arrayCycle (<collection>[,n=100]); arrayRepeat (<value>[,n=100]); iterRange ([start=0[,step=1[,end=Infinity]]]); iterCycle (<iter>[,n=Infinity]); iterRepeat (<value>[,n=Infinity]); arrayUnion (<collection1>[,collectionN]); arrayIntersection (<collection1>,<collection2>); arrayDifference (<collection1>,<collection2>); arraySymmetricDifference (<collec1>,<collec2>); setUnion (<collection1>[,collectionN]); setIntersection (<set1>,<set2>); setDifference (<set1>,<set2>); setSymmetricDifference (<set1>,<set2>); isSuperset (<superCollection>,<subCollection>); slice (<collection>[,begin=0[,end=Infinity]]); without (<collection>,<filterCollection>); reduce (<collection>,<callback>[,initialvalue]); take (<collection>[,n=1]); takeWhile (<collection>,<callback>); takeRight (<collection>[,n=1]); takeRightWhile (<collection>,<callback>); drop (<collection>[,n=1]); dropWhile (<collection>,<callback>); dropRight (<collection>[,n=1]); dropRightWhile (<collection>,<callback>);	forEach(<collection>,<callback>); map(<collection>,<callback>); enumerate(<collection>[,offset=0]); entries(<collection>[,offset=0]); size(<collection>); every(<collection>,<callback>); some(<collection>,<callback>); none(<collection>,<callback>); includes(<collection>,<value>); contains(<collection>,<value>); find(<collection>,<callback>); findLast(<collection>,<callback>); filter(<collection>,<callback>); reject(<collection>,<callback>); partition(<collection>,<callback>); groupBy(<collec.>,<cb>[,map=false]); shuffle(<collection>); min(<value1>[,valueN]); max(<value1>[,valueN]); sort(<collection>[,numbers=false]); reverse(<collection>); zip(<collection1>[,collectionN]); unzip(<collection>); zipObj (<collection1>,<collection2>); item(<collection>,<index>); nth(<collection>,<index>); first(<collection>); head(<collection>); last(<collection>); initial(<collection>); tail(<collection>); flat(<collection>); concat (<collection1>[,collectionN]); join (<collection>[,separator=","]);	Array.prototype.at(); Array.prototype.findLast(); Array.prototype.findLastIndex(); Array.prototype.flat(); + .flatMap(); Array.prototype.groupBy(<fn>); Array.prototype.groupByToMap(<fn>); crypto.randomUUID();, globalThis; Object.hasOwn(); and .fromEntries(); String.prototype.at(); + .matchAll(); String.prototype.replaceAll(); String.prototype.trimStart(); String.prototype.trimLeft(); String.prototype.trimEnd(); String.prototype.trimRight(); TypedArray.prototype.at(); TypedArray.prototype.findLast(); TypedArray.prototype.findLastIndex();
		Non-standard polyfills
		BigInt.prototype.toJSON(); AsyncFunction(); GeneratorFunction();
		Abstract functions
		getInV(<o>,<pr>);, toInteger(<val>); hasIn(<obj>,<prop>); getIn(<o>,<pr>); setIn(<object>,<property>,<value>); isPropertyKey(<v>); and isIndex(<v>); toPropertyKey(<v>); and toIndex(<v>); toObject(<value>); and type(<value>); createDataProperty(<obj>,<prop>,<v>); createMethodProperty(<obj>,<pr>,<v>); isSameValue(<v1>,<v2>);, toArray(v); isSameValueZero(<value1>,<value2>); isSameValueNonNumber (<val1>,<val2>);
Cookie		
getCookie ([name]);, hasCookie (<name>);, setCookie (<name>,<value>[,hours=8760[,path="/"[,domain[,secure[,SameSite="Lax"[,HttpOnly]]]]]]);, setCookie (<Optionsobj>); removeCookie (<name>[,path="/"[,domain[,secure[,SameSite="Lax"[,HttpOnly]]]]]);, removeCookie (<Options object>);, clearCookies ([path="/"[,domain[,sec[,SameSite="Lax"[,HttpOnly]]]]]);, clearCookies (<Options object>);		