# Celestra cheatsheet – v5.3.1 – 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 |
|---|---|---|
| signbit(<value>);<br>delay(<ms>).then(<callback>);<br>inherit(<subclass>,<superclass>);<br>randomInt([max] *or* <min>,<max>);<br>randomFloat([max] *or* <min>,<max>);<br>randomBoolean();<br>randomID([hyphens=true][,usedate=false]);<br>randomString([length[,specChar=false]]);<br>inRange(<value>,<min>,<max>);<br>b64Encode(<string>);, b64Decode(<str>);<br>javaHash(<data>[,hexa=false]);<br>getUrlVars([str=location.search]);<br>obj2string(<object>);<br>getType(<variable>[,type][,throw=false]);<br>extend([deep,]<target>,<source1>[,srcN]);<br>sizeIn(<obj>); *and* forIn(<obj>,<cb>);<br>filterIn(<obj>,<cb>);, popIn(<obj>,<pr>);<br>strPropercase(<s>);, strCapitalize(<s>);<br>strUpFirst(<str>);, strDownFirst(<str>);<br>strHTMLRemoveTags(<string>);<br>strHTMLEscape(<s>);,strHTMLUnEscape(<s>);<br>strReverse(<str>);, strAt(<str>,<index>);<br>strCodePoints(<string>);<br>strFromCodePoints(<collection>);<br>toFunction(<fn>);, bind(<fn>,<context>);<br>constant(<value>); *and* identity(<value>);<br>noop(); *and* T(); *and* F();<br>assertEq(<msg>,<v1>,<v2>[,strict=true]);<br>assertNotEq(<m>,<v1>,<v2>[,strict=true]);<br>assertTrue(<msg>,<value>);<br>assertFalse(<msg>,<value>);<br>noConflict(); *and* VERSION; | qsa(<selector>[,context]).forEach(<cb>);<br>qs(<selector>[,context]);<br>domReady(<callback>);<br>domCreate(<type>[,properties[,innerHTML]]);<br>domCreate(<element descriptive object>);<br>domToElement(<htmlString>);<br>domGetCSS(<element>[,property]);<br>domSetCSS(<element>,<property>,<value>);<br>domSetCSS(<element>,<properties>);<br>domFadeIn(<element>[,duration[,display]]);<br>domFadeOut(<element>[,duration]);<br>domFadeToggle(<elem.>[,duration[,display]]);<br>domShow(<element>[,display]);<br>domHide(<element>);<br>domToggle(<element>[,display]);<br>domIsHidden(<element>);<br>domSiblings(<element>);<br>domSiblingsPrev(<element>);<br>domSiblingsLeft(<element>);<br>domSiblingsNext(<element>);<br>domSiblingsRight(<element>);<br>domGetCSSVar(<name>);<br>domSetCSSVar(<name>,<value>);<br>importScript(<script1>[,scriptN]);<br>importStyle(<style1>[,styleN]);<br>setFullscreenOn(<selector> *or* <element>);<br>setFullscreenOff();<br>getFullscreen();<br>form2array(<form>); *and* form2string(<form>);<br>getDoNotTrack();<br>getLocation(<success>[,error]);<br>createFile(<filename>,<content>[,dType]); | isMap(<value>); *and* isWeakMap(<v>);<br>isSet(<value>); *and* isWeakSet(<v>);<br>isNumber(<v>); *and* isNumeric(<v>);<br>isFloat(<v>); *and* isBigInt(<v>);<br>isString(<v>); *and* isChar(<v>);<br>isDate(<v>); *and* isError(<v>);<br>isRegexp(<v>); *and* isSymbol(<v>);<br>isElement(<v>); *and* isObject(<v>);<br>isDataView(<v>);, isBoolean(<v>);<br>isNull(<val>);, isUndefined(<val>);<br>isNullOrUndefined(<v>); isNil(<v>);<br>isPlainObject(<value>);<br>isFunction(<value>);<br>isConstructorFn(<value>);<br>isGeneratorFn(<value>);<br>isAsyncFn(<value>);<br>isArraylike(<value>);<br>isTypedArray(<value>);<br>isArrayBuffer(<value>);<br>isPrimitive(<value>);<br>isIterator(<v>);, isIterable(<v>);<br>isPromise(<value>);<br>isEmptyObject(<value>);<br>isEmptyArray(<value>);<br>isEmptyMap(<value>);<br>isEmptySet(<value>);<br>isEmptyIterator(<value>);<br>isSameObject(<object1>,<object2>);<br>isSameArray(<array1>,<array2>);<br>isSameMap(<map1>,<map2>);<br>isSameSet(<set1>,<set2>);<br>isSameIterator(<iter1>,<iter2>); |

**AJAX and CORS**

ajax(<Options object>);, getJson(<url>,<success>);, getText(<url>,<success>);
**Options object properties (* = default value):** url: *string*, data: *string*, queryType: **"ajax"/"cors"*, type: **"get"/"post"*, success: *function*, error: *function*, format: **"text"/"json"/"xml"*, user: *string*, password: *string*

## Collections

```
arrayMerge([flat=false,]<target>,<src1>[,srN]);
arrayUnique(<collection>);
arrayAdd(<array>,<value>);
arrayClear(<array>);
arrayRemove(<array>,<value>[,all=false]);
arrayRemoveBy(<array>,<callback>[,all=false]);
arrayRange([start=0[,end=100[,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]);
shuffle(<collection>);
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(<collection>,<callback>);

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=","]);
```

## Polyfills

```
Array.prototype.at();
Array.prototype.findLast();
Array.prototype.findLastIndex();
Array.prototype.flat(); + .flatMap();
crypto.randomUUID();
globalThis;
Object.hasOwn(); and .fromEntries();
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();
```

## Non-standard polyfills

```
BigInt.prototype.toJSON();
window.AsyncFunction();
window.GeneratorFunction();
```

## Abstract functions

```
hasIn(<obj>,<prop>); getIn(<o>,<pr>);
setIn(<object>,<property>,<value>);
isPropertyKey(<v>); and isIndex(<v>);
toPropertyKey(<v>); and toIndex(<v>);
toInteger(<value>);
toObject(<value>); and type(<value>);
isSameValueZero(<value1>,<value2>);
createMethodProperty(<obj>,<pr>,<v>);
```

## 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>);
```