Celestra cheatsheet – v3.6.1 – 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
inherit(<subclass>,<superclass>);	qsa(<selector>[,context]).forEach(<fn>);	isString(<value>);
randomInt([max] *or* <min>,<max>);	qs(<selector>[,context]).argument;	isChar(<value>);
randomFloat([max] *or* <min>,<max>);	domReady(<function>);	isNumber(<value>);
randomString([length[,specChar]]);	domCreate(<type>[,properties[,innerHTML]]);	isNumeric(<value>);
b64Encode(<string>);	domCreate(<element descriptive object>);	isFloat(<value>);
b64Decode(<string>);	domToElement(<htmlString>);	isBigInt(<value>);
javaHash(<data>[,hexa]);	domGetCSS(<element>,<property>);	isDate(<value>);
getUrlVars([str=location.search]);	domSetCSS(<element>,<property>,<value>);	isBoolean(<value>);
obj2string(<object>);	domSetCSS(<element>,<properties>);	isElement(<value>);
getType(<variable>[,type]);	domFadeIn(<element>[,duration[,display]]);	isObject(<value>);
extend([deep,]<target>,<source1>[,srcN]);	domFadeOut(<element>[,duration]);	isEmptyObject(<value>);
deepAssign(<target>,<source1>[,srcN]);	domFadeToggle(<elem.>[,duration[,display]]);	isArraylike(<value>);
hasOwn(<object>,<property>);	domShow(<element>[,display]);	isSameArray(<array1>,<array2>);
forIn(<object>,<callback>);	domHide(<element>);	isEmptyArray(<value>);
strRemoveTags(<string>);	domToggle(<element>[,display]);	isTypedArray(<value>);
strReverse(<string>);	domIsHidden(<element>);	isArrayBuffer(<value>);
strReplaceAll(<str>,<search>,<replace>);	domSiblings(<element>);	isNull(<value>);
strCodePoints(<string>);	domGetCSSVar(<name>);	isUndefined(<value>);
strFromCodePoints(<iter>);	domSetCSSVar(<name>,<value>);	isNullOrUndefined(<value>);
strAt(<string>,<pos>);	importScript(<url>[,success]);	isNil(<value>);
toFunction(<function>);	importScripts(<scripts> *or* <script1>[,scN]);	isPrimitive(<value>);
bind(<function>,<context>);	importStyle(<href>[,success]);	isRegexp(<value>);
constant(<value>);	importStyles(<styles> *or* <style1>[,styleN]);	isSymbol(<value>);
identity(<value>);	setFullscreenOn(<selector> *or* <element>);	isIterator(<value>);
noop(); *and* T(); *and* F();	setFullscreenOff();	isIterable(<value>);
assert(<condition>[,message]);	getFullscreen();	isFunction(<value>);
assertLog(<condition>[,message]);	form2array(<form>); *and* form2string(<form>);	isGeneratorFn(<value>);
assertAlert(<condition>[,message]);	getDoNotTrack();	isAsyncFn(<value>);
noConflict();	getLocation(<success>[,error]);	isMap(<v>); *and* isWeakMap(<v>);
VERSION;	createFile(<filename>,<content>[,dType]);	isSet(<v>); *and* isWeakSet(<v>);

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*

Cookie

setCookie(<name>,<value>[,hours[,path[,domain[,secure[,SameSite[,HttpOnly]]]]]]);, getCookie([name]);, hasCookie(<name>);, removeCookie(<name>[,path[,domain[,secure[,SameSite[,HOnly]]]]]);, clearCookies([path[,domain[,sec[,SameSite[,HOnly]]]]]);

Collections	Polyfills
`isSuperset(<superset>,<subset>);`	`Array.prototype.values();`
`arrayMerge([deep,]<target>,<source1>[,srcN]);`	`Array.prototype.includes();`
`zip(<collection1>[,collectionN]); and unzip(<collection>);`	`Array.prototype.flat();`
`uniqueArray(<value>); and uniquePush(<array>,<value>);`	`Array.prototype.flatMap();`
`arrayClear(<array>); and arrayRemove(<array>,<value>[,all]);`	`String.prototype.includes();`
`min(<collection>); and minIndex(<collection>);`	`String.prototype.trimStart();`
`max(<collection>); and maxIndex(<collection>);`	`String.prototype.trimLeft();`
`setUnion(<collection1>[,collectionN]);`	`String.prototype.trimEnd();`
`setIntersection(<set1>,<set2>);`	`String.prototype.trimRight();`
`setDifference(<set1>,<set2>);`	`String.prototype.padStart();`
`setSymmetricDifference(<set1>,<set2>);`	`String.prototype.padEnd();`
`arrayUnion(<collection1>[,collectionN]);`	`String.prototype.repeat();`
`arrayIntersection(<collection1>,<collection2>);`	`String.prototype.replaceAll();`
`arrayDifference(<collection1>,<collection2>);`	`String.prototype.matchAll();`
`arraySymmetricDifference(<collection1>,<collection2>);`	`String.prototype[Symbol.iterator]();`
`arrayRange([start=0[,end=100[,step=1]]]); and iterRange([start=0[,step=1[,end]]]);`	`Object.assign();`
`arrayCycle(<collection>[,n]); and iterCycle(<iter>[,n]);`	`Object.fromEntries();`
`arrayRepeat(<value>[,n]); and iterRepeat(<value>[,n]);`	`Object.entries();`
`sizeOf(<collection>);`	`Object.values();`
`item(<collection>,<index>); and itemOf(<collection>,<index>);`	`Object.getOwnPropertyDescriptors();`
`forOf(<collection>,<callback>); and forEach(<collection>,<callback>);`	`RegExp.prototype.flags;`
`mapOf(<collection>,<callback>); and map(<collection>,<callback>);`	`NodeList.prototype.forEach();`
`filterOf(<collection>,<callback>);`	`ChildNode.after();`
`hasOf(<collection>,<value>);`	`ChildNode.before();`
`findOf(<collection>,<callback>);`	`ChildNode.remove();`
`everyOf(<collection>,<callback>); and someOf(<collection>,<callback>);`	`ChildNode.replaceWith();`
`noneOf(<collection>,<callback>);`	`ParentNode.append();`
`firstOf(<collection>); and lastOf(<collection>);`	`ParentNode.prepend();`
`sliceOf(<collection>[,begin[,end]]);`	`Element.prototype.matches();`
`reverseOf(<collection>);`	`Element.prototype.closest();`
`sortOf(<collection>);`	`Element.prototype.toggleAttribute();`
`reduceOf(<collection>,<callback>[,initialvalue]);`	`Element.prototype.getAttributeNames();`
`concatOf(<collection1>[,collectionN]); and flatOf(<collection>);`	`window.screenLeft;`
`enumerateOf(<collection>);`	`window.screenTop;`
`joinOf(<collection>[,separator]);`	`globalThis;`
`takeOf(<collection>[,n]); and takeWhile(<collection>,<callback>);`	`BigInt.prototype.toJSON();`
`takeRight(<collection>[,n]); and takeRightWhile(<collection>,<callback>);`	`window.GeneratorFunction();`
`dropOf(<collection>[,n]); and dropWhile(<collection>,<callback>);`	`window.AsyncFunction();`
`dropRight(<collection>[,n]); and dropRightWhile(<collection>,<callback>);`	