Celestra v1.16.1 cheatsheet – https://github.com/Serrin/Celestra/

Basic API

```
doc = document
qsa(<selector>[,context]);
qsa(<selector>[,context]).each( fn (el,i)
{ el.arguments;} );
qs(<selector>[,context]).argument;
domReady(<fn>);
inherit(<subclass>,<superclass>);
random(<max>);
random(<min>,<max>);
getScript(<url>[,success]);
getScripts(<scripts>);
getStyle(<href>[,success]);
getStyles(<styles>);
getUrlVar([name]);
getUrlVarFromString(<querystr>[,name]);
obj2string(<object>);
getType(<variable>[, type]);
extend([deep,]<target>,<source1>,
...sources);
getFullscreen();
setFullscreenOn(<selector>);
setFullscreenOn(<element>);
setFullscreenOff();
getLocation(<success>[,error]);
getDoNotTrack();
constant(<value>);
identity(<value>);
noop();
repeat(<iteration>,<callback>);
```

DOM

```
domCreate(<type>[,properties[,innerHTML]]);
domGetCSS(<element>,<property>);
domSetCSS(<element>,<property>,<value>);
domSetCSS(<element>,<properties>);
domFadeIn(<element>[, duration[,display]]);
domFadeOut(<element>[,duration]);
domFadeToggle(<element>[,duration[,display]]);
domShow(<element>[,display]);
domHide(<element>);
domToggle(<element>[,display]);
domOn(<eventTarget>,<eventType>,<callback>);
domOff(<eventTarget>,<eventType>,<callback>);
domTrigger(<eventTarget>,<eventType>);
```

Type checking

```
isString(<value>);
isChar(<value>);
isNumber(<value>);
isInteger(<value>);
isFloat(<value>);
isBoolean(<value>);
isObject(<value>);
isEmptyObject(<value>);
isFunction(<value>);
isArray(<value>);
isEmptyArray(<value>);
isArraylike(<value>);
isNull(<value>);
isUndefined(<value>);
isNullOrUndefined(<value>);
isPrimitive(<value>);
isSymbol(<value>); ES6
isMap(<value>); ES6
isSet(<value>); Es6
```

AJAX and CORS

```
getJson (<url>,<success>);
getText (<url>,<success>);
getAjax (<url>,<format>,<success>[,error]
[,user<,password>]);
postAjax (<url>,<data>,<format>,<success>
[,error][,user<,password>]);
getCors (<url>,<format>,<success>[,error]
[,user<,password>]);
postCors (<url>,<data>,<format>,<success>
[,error][,user<,password>]);
```

Functional programming

```
toFunction(<function>);
bind(<function>,<context>);
forEach(<collection>,<callback>);
map(<collection>,<callback>);
forIn(<object>,<callback>);
mapIn(<object>,<callback>);
toArray(<object>);
toObject(<array>);
```

Polyfills

```
Array.from(), Array.of(), Object.create(), Object.assign(), ChildNode.after(), ChildNode.before(), ChildNode.remove(),
ChildNode.replaceWith(), ParentNode.append(), ParentNode.prepend(), Array.prototype.includes(), String.prototype.includes(),
NodeList.prototype.forEach(), Number.MIN_SAFE_INTEGER, Number.MAX_SAFE_INTEGER, Number.EPSILON, Number.isNaN(), isNaN(),
Number.isInteger(), Number.isFinite(), Number.isSafeInteger()
```