

Celestra v1.16.2 cheatsheet – <https://github.com/Serrin/Celestra/>

Basic API	DOM	Type checking
<code>doc = document</code> <code>qsa(<selector>[, context]);</code> <code>qsa(<selector>[, context]).each(fn (el, i) { el.arguments; });</code> <code>qs(<selector>[, context]).argument;</code> <code>domReady(<fn>);</code> <code>inherit(<subclass>, <superclass>);</code> <code>random(<max>);</code> <code>random(<min>, <max>);</code> <code>getScript(<url>[, success]);</code> <code>getScripts(<scripts>);</code> <code>getStyle(<href>[, success]);</code> <code>getStyles(<styles>);</code> <code>getUrlVar([name]);</code> <code>getUrlVarFromString(<querystr>[, name]);</code> <code>obj2string(<object>);</code> <code>getType(<variable>[, type]);</code> <code>extend([deep,]<target>, <source1>, ...sources);</code> <code>getFullscreen();</code> <code>setFullscreenOn(<selector>);</code> <code>setFullscreenOn(<element>);</code> <code>setFullscreenOff();</code> <code>getLocation(<success>[, error]);</code> <code>getDoNotTrack();</code> <code>constant(<value>);</code> <code>identity(<value>);</code> <code>noop();</code> <code>repeat(<iteration>, <callback>);</code>	<code>domCreate(<type>[, properties[, innerHTML]]);</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(<element>[, duration[, display]]);</code> <code>domShow(<element>[, display]);</code> <code>domHide(<element>);</code> <code>domToggle(<element>[, display]);</code> <code>domOn(<eventTarget>, <eventType>, <callback>);</code> <code>domOff(<eventTarget>, <eventType>, <callback>);</code> <code>domTrigger(<eventTarget>, <eventType>);</code>	<code>isString(<value>);</code> <code>isChar(<value>);</code> <code>isNumber(<value>);</code> <code>isInteger(<value>);</code> <code>isFloat(<value>);</code> <code>isBoolean(<value>);</code> <code>isObject(<value>);</code> <code>isEmptyObject(<value>);</code> <code>isFunction(<value>);</code> <code>isArray(<value>);</code> <code>isEmptyArray(<value>);</code> <code>isArraylike(<value>);</code> <code>isNull(<value>);</code> <code>isUndefined(<value>);</code> <code>isNullOrUndefined(<value>);</code> <code>isPrimitive(<value>);</code> <code>isSymbol(<value>); ES6</code> <code>isMap(<value>); ES6</code> <code>isSet(<value>); ES6</code>
	AJAX and CORS	Functional programming
	<code>getJSON (<url>, <success>);</code> <code>getText (<url>, <success>);</code> <code>getAjax (<url>, <format>, <success>[, error] [, user<, password>]);</code> <code>postAjax (<url>, <data>, <format>, <success> [, error] [, user<, password>]);</code> <code>getCors (<url>, <format>, <success>[, error] [, user<, password>]);</code> <code>postCors (<url>, <data>, <format>, <success> [, error] [, user<, password>]);</code>	<code>toFunction(<function>);</code> <code>bind(<function>, <context>);</code> <code>forEach(<collection>, <callback>);</code> <code>each(<collection>, <callback>);</code> <code>map(<collection>, <callback>);</code> <code>forIn(<object>, <callback>);</code> <code>mapIn(<object>, <callback>);</code> <code>toArray(<object>);</code> <code>toObject(<array>);</code>
Polyfills		
<code>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()</code>		