Celestra v1.18.3 FP cheatsheet – https://github.com/Serrin/Celestra/

Basic API

```
doc = document
qsa(<selector>[,context]).each( fn (el,i)
{ el.arguments;} );
qs(<selector>[,context]).argument;
domReady(<fn>);
inherit(<subclass>,<superclass>);
random(<max>); or 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>[,srcN]);
getFullscreen();
setFullscreenOn(<selector> or <element>);
setFullscreenOff();
getLocation(<success>[,error]);
getDoNotTrack();
form2array(<form>);
form2string(<form>);
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>);
```

Functional programming

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

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>]);
```

Cookie

```
setCookie(<name>,<value>[,hours[,path[,domain[,secure[,HttpOnly]]]]]);
getCookie([name]);
hasCookie(<name>);
removeCookie(<name>[,path[,domain[,secure[,HttpOnly]]]]);
```

initType

```
initArray();, initObject();, initString();, initTrue();, initFalse();
```

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, isDate(<value>);, isRegexp(<value>);, isElement(<value>);
```

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();
```