JavaScript cheatsheet – v3.8.1 – https://github.com/Serrin/Celestra/

```
Web Storage api and JSON
                                          element.dataset & data-* attributes
                                                                                                  TypedArray
IE8+
                                          - IE11 compatible
                                                                                IE10+11 compatible
                                          - element data-* attributes
localStorage:
                                          - no methods and events
                                                                                new <TypedArray>(); ES2017
localStorage.length;
                                                                               new <TypedArray>(length);
localStorage.kev(index);
                                          camelcase:
                                                                                new <TypedArray>(typedArray);
localStorage.getItem(key);
                                          element.data-name
                                                                                new <TypedArray>(object);
localStorage.setItem(key, data);
                                            -> element.dataset.name
                                                                                new <TypedArray>(buffer[,byteOffset[,len]]);
localStorage.removeItem(key);
                                          element.data-first-second
localStorage.clear();
                                            -> element.dataset.firstSecond
                                                                                Int8Arrav();
                                                                                -128 to 127, 1 byte, int8 t, Shortint
                                                                                Uint8Array();
sessionStorage:
                                          set:
sessionStorage.length;
                                          element.dataset.name = "value";
                                                                                0 to 255, 1 byte, uint8 t, Byte
sessionStorage.kev(index);
                                          element.dataset["name"] = "value";
                                                                                Uint8ClampedArray(); - not in IE10-11
sessionStorage.getItem(key);
                                          element.setAttribute("data-name",
                                                                               0 to 255, 1 byte, uint8 t, Byte
sessionStorage.setItem(key, data);
                                          "value"):
                                                                                Int16Array();
sessionStorage.removeItem(key);
                                          element["data-name"] = "value";
                                                                                -32768 to 32767, 2 byte, int16 t, Smallint
sessionStorage.clear();
                                                                                Uint16Array();
                                                                                0 to 65535, 2 byte, uint16 t, Word
                                          element.dataset.name;
hasItem:
                                                                                Int32Array();
localStorage.getItem(key) !== null
                                          element.dataset["name"];
                                                                                -2147483648 to 2147483647, 4 byte, int32 t
sessionStorage.getItem(key) !== null
                                          element.getAttribute("data-name");
                                                                               Uint32Array();
                                          element["data-name"];
                                                                                0 to 4294967295, 4 byte, uint32 t, Longword
setJSON:
                                                                               BigInt64Array(); - not in IE10-11
                                                                                -2**63 to 2**63-1, 8 byte, int64 t, Int64
localStorage.setItem(key,
                                          remove:
JSON.stringify(object));
                                                                               BigUint64Array(); - not in IE10-11
                                          element.removeAttribute("data-
                                                                                0 to 2**64-1, 8 byte, uint64 t, QWord
sessionStorage.setItem(key,
                                          name");
JSON.stringify(object));
                                                                               Float32Array();
                                          check:
                                                                               1.2x10-38 to 3.4x1038, 4 byte, float, Real
getJSON:
                                          element.hasAttribute("data-name");
                                                                                Float64Array();
JSON.parse(localStorage.getItem(key));
                                                                                5.0 \times 10-324 to 1.8 \times 10308, 8 byte, Double
JSON.parse(sessionStorage.getItem(key));
                                                         DOM events
target.addEventListener(<type>,<listener>[,useCapture]); or target.addEventListener(<type>,<listener>[,options]);
target.removeEventListener(<type>,<listener>[,useCapture]); or target.removeEventListener(<type>,<listener>[,options]);
target.dispatchEvent(<event>);
target.type(); or target["type"]();
```

```
element.classList
                                                                                           JSON
IE10+IE11 don't have support for classList on SVG or MathML|IE8+
elements.
                                                                Valid Data Types
element.classList.add(String[,String]);
                                                                - string
IE10+11: ves (except the multiple arguments)
                                                                - number
                                                                - object (containing valid JSON values)
element.classList.remove(String[,String]);
                                                                - arrav
IE10+11: yes (except the multiple arguments)
                                                                - boolean
- Removing a class that does not exist, does NOT throw an - date
error.
                                                                - null
element.classList.contains(String);
                                                                Invalid Data Types
                                                                - function
IE10+11: yes
                                                                - Symbol
element.classList.toggle(String[,force]);
                                                                - NaN, Infinity, undefined - will be "null"
IE10+11: yes (except the second argument)
                                                                - an object with method(s) (functions)
- When only one argument is present: Toggle class value; if - Map, Set, WeakMap, WeakSet - fix: convert to array
class exists then remove it and return false, if not, then add - BigInt - fixed in Celestra - BigInt.prototype.toJSON();
it and return true.
- When a second argument is present: If the second argument | JSON.stringify(value[,replacer[,space]]);
evaluates to true, add specified class value, and if it Convert a JavaScript object to a JSON string.
evaluates to false, remove it.
                                                                JSON.stringify( { a: 1, b: "2", c: true } );
                                                                // -> "{\"a\":1,\"b\":\"2\",\"c\":true}"
element.classList.item(Number);
IE10+11: yes
                                                                JSON.stringify([1, 2, 3, 4, 5]);
element.classList.length;
                                                                // -> "[1,2,3,4,5]"
IE10+11: yes
                                                                JSON.parse(text[,reviver]);
element.classList.replace(oldClass, newClass);
                                                                Parses a JSON string and returns a JavaScript object.
IE10+11: No and the method isn't compatible with the Safari and
mobile browsers too.
                                                                JSON.parse(JSON.stringify( {a: 1, b: "2", c: true} ));
                                                                // -> Object { a: 1, b: "2", c: true }
Remove all classes:
element.className = "";
                                                                JSON.parse(JSON.stringify([1, 2, 3, 4, 5]));
                                                                // -> Array(5) [ 1, 2, 3, 4, 5 ]
```

```
BigInt (Int64)
                           DOMParser
IE9: XML support
                                                               let theBiggestInt = 9007199254740991n;
IE10+IE11: XML, SVG and HTML support
                                                               let alsoHuge = BigInt(9007199254740991);
                                                               let hugeString = BigInt("9007199254740991");
var parser = new DOMParser();
                                                               let hugeHex = BigInt("0x1ffffffffffffff");
var doc = parser.parseFromString("sourceStr", "application/xml");
                                                               let hugeBin =
Returns a Document, but not a SVGDocument nor a HTMLDocument.
                                                               1111111");
var parser = new DOMParser();
                                                                // all -> 9007199254740991n
var doc = parser.parseFromString(sourceStr, "image/svg+xml");
Returns a SVGDocument, which also is a Document.
                                                                typeof 1n;
                                                                            // -> "bigint"
                                                               typeof BigInt("1"); // -> "bigint"
var parser = new DOMParser();
var doc = parser.parseFromString(sourceStr, "text/html");
                                                               Operators: +, *, -, **, %, Bitwise operators (e.g.: >>>)
Returns a HTML document.
                                                                !0n
                                                                       // -> true
                                                                        // -> false
                                                                l1n
                   DOMParser sample function
                                                               4n / 2n / / -> 2n
                                                               5n / 2n // -> 2n, not 2.5n -> rounded
function parseHTML (str) {
 return Array.from(
                                                               1n < 2 // -> true
    (new DOMParser())
                                                               2n > 1 // -> true
     .parseFromString(str, "text/html")
                                                               2n > 2 // -> false
     .childNodes[0]
                                                               2n >= 2 // -> true
     .childNodes[1]
     .childNodes
                                                               let mixed = [4n, 6, -12n, 10, 4, 0, 0n];
 );
                                                               mixed.sort();// \rightarrow [-12n, 0, 0n, 4n, 4, 6, 10]
                                                               Methods
parseHTML(
 "<div>123<"
                                                               BigInt.asIntN();
 + "<div>456</div>"
                                                                 - BigInt value to a signed integer
 + "<div>7</div>"
                                                               BigInt.asUintN();
                                                                 - BigInt value to an unsigned integer
 + "8"
                                                               BigInt.prototype.toLocaleString();
// -> Array(4) [ div, div, div, p ]
                                                               BigInt.prototype.toString();
// Tested in IE11, Edge, Firefox and Chrome.
                                                               BigInt.prototype.valueOf();
```

```
Fetch
                                                                                            Fetch POST
Firefox 39, Chrome 42, Edge14, Opera29, Safari 10.1, Samsung I. 4.0
                                                                       // Example POST method implementation:
                                                                       // Default options are marked with *
// Example GET method implementation with TEXT:
                                                                       async function postData(url = "url", data = {}) {
fetch("https://api.coindesk.com/v1/bpi/currentprice.json")
                                                                         const response = await fetch(url, {
  .then( response => response.text() )
                                                                           method: "POST",
  .then( data => console.log(data) )
                                                                           // *GET, POST, PUT, DELETE, etc.
  .catch( error => console.log(error) );
                                                                           mode: "cors",
                                                                           // no-cors, *cors, same-origin
// Example GET method implementation with JSON:
                                                                           cache: "no-cache",
fetch("https://api.coindesk.com/v1/bpi/currentprice.json")
                                                                           // *default, no-cache, reload, force-cache,
  .then( response => response.json() )
                                                                       only-if-cached
  .then( data => console.log(data.bpi.USD.rate) )
                                                                           credentials: "same-origin",
  .catch( error => console.log(error) );
                                                                           // include, *same-origin, omit
                                                                           headers: {"Content-Type": "application/json"},
// Example GET method implementation with TEXT and JSON:
                                                                           // "Content-Type": "application/x-www-form-
fetch("https://api.coindesk.com/v1/bpi/currentprice.json")
                                                                       urlencoded"
  .then( response => response.text() )
                                                                           redirect: "follow",
  .then(text => console.log(JSON.parse(text).bpi.USD.rate+"\n"+text))
                                                                           // manual, *follow, error
  .catch( error => console.log(error) );
                                                                           referrerPolicy: "no-referrer",
                                                                           // no-referrer, *no-referrer-when-downgrade,
// Example POST method implementation with upload JSON data:
                                                                       origin, origin-when-cross-origin, same-origin,
const data = { username: "example" };
                                                                       strict-origin, strict-origin-when-cross-origin,
fetch("https://example.com/profile", {
                                                                       unsafe-url
   method: "POST", // or "PUT"
                                                                           body: JSON.stringify(data)
   headers: { "Content-Type": "application/json", },
                                                                           // body data type must match "Content-Type"
   body: JSON.stringify(data),
                                                                       header
                                                                         });
  .then(response => response.json())
                                                                         return response.json();
  .then(data => { console.log("Success:", data); })
                                                                        // parses JSON response into native JavaScript
  .catch((error) => { console.error("Error:", error); });
                                                                       objects
                                                                       postData("https://example.com/answer", { answer:
                                                                       42 })
                                                                         .then(data => { console.log(data); });
                                                                         // JSON data parsed by `data.json()` call
```