

JavaScript cheatsheet – v3.4.1 – <https://github.com/Serrin/Celestra/>

| Web Storage api and JSON | element.dataset & data-* attributes | TypedArray |
|--|---|--|
| <p>IE8+</p> <p>localStorage: localStorage.length; localStorage.key(index); localStorage.getItem(key); localStorage.setItem(key, data); localStorage.removeItem(key); localStorage.clear();</p> <p>sessionStorage: sessionStorage.length; sessionStorage.key(index); sessionStorage.getItem(key); sessionStorage.setItem(key, data); sessionStorage.removeItem(key); sessionStorage.clear();</p> <p>hasItem: localStorage.getItem(key) !== null sessionStorage.getItem(key) !== null</p> <p>setJSON: localStorage.setItem(key, JSON.stringify(object)); sessionStorage.setItem(key, JSON.stringify(object));</p> <p>getJSON: JSON.parse(localStorage.getItem(key)); JSON.parse(sessionStorage.getItem(key));</p> | <p>- IE11 compatible - element data-* attributes - no methods and events</p> <p>camelcase: element.data-name -> element.dataset.name element.data-first-second -> element.dataset.firstSecond</p> <p>set: element.dataset.name = "value"; element.dataset["name"] = "value"; element.setAttribute("data-name", "value"); element["data-name"] = "value";</p> <p>get: element.dataset.name; element.dataset["name"]; element.getAttribute("data-name"); element["data-name"];</p> <p>remove: element.removeAttribute("data-name");</p> <p>check: element.hasAttribute("data-name");</p> | <p>IE10+11 compatible</p> <p>new <TypedArray>(); <i>ES2017</i> new <TypedArray>(length); new <TypedArray>(typedArray); new <TypedArray>(object); new <TypedArray>(buffer[,byteOffset[,len]]);</p> <p>Int8Array(); -128 to 127, 1 byte, int8_t</p> <p>Uint8Array(); 0 to 255, 1 byte, uint8_t</p> <p>Uint8ClampedArray(); - not in IE10-11 0 to 255, 1 byte, uint8_t</p> <p>Int16Array(); -32768 to 32767, 2 byte, int16_t</p> <p>Uint16Array(); 0 to 65535, 2 byte, uint16_t</p> <p>Int32Array(); -2147483648 to 2147483647, 4 byte, int32_t</p> <p>Uint32Array(); 0 to 4294967295, 4 byte, uint32_t</p> <p>BigInt64Array(); - not in IE10-11 -2**63 to 2**63-1, 8 byte, int64_t</p> <p>BigUint64Array(); - not in IE10-11 0 to 2**64-1, 8 byte, uint64_t</p> <p>Float32Array(); 1.2x10-38 to 3.4x1038, 4 byte, float</p> <p>Float64Array(); 5.0x10-324 to 1.8x10308, 8 byte, double</p> |
| DOM events | | |
| <p>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"]();</p> | | |

| element.classList | JSON |
|---|--|
| <p>IE10+IE11 don't have support for classList on SVG or MathML elements.</p> <p>element.classList.add(String[,String]); IE10+11: yes (except the multiple arguments)</p> <p>element.classList.remove(String[,String]); IE10+11: yes (except the multiple arguments) - Removing a class that does not exist, does NOT throw an error.</p> <p>element.classList.contains(String); IE10+11: yes</p> <p>element.classList.toggle(String[,force]); IE10+11: yes (except the second argument) - When only one argument is present: Toggle class value; if class exists then remove it and return false, if not, then add it and return true. - When a second argument is present: If the second argument evaluates to true, add specified class value, and if it evaluates to false, remove it.</p> <p>element.classList.item(Number); IE10+11: yes</p> <p>element.classList.length; IE10+11: yes</p> <p>element.classList.replace(oldClass, newClass); IE10+11: No and the method isn't compatible with the Safari and mobile browsers too.</p> <p>Remove all classes: element.className = "";</p> | <p>IE8+</p> <p>Valid Data Types</p> <ul style="list-style-type: none"> - string - number - object (containing valid JSON values) - array - boolean - date - null <p>Invalid Data Types</p> <ul style="list-style-type: none"> - function - Symbol - NaN, Infinity, undefined - <i>will be "null"</i> - an object with method(s) (functions) - Map, Set, WeakMap, WeakSet - <i>fix: convert to array</i> - BigInt - <i>fixed in Celestra - BigInt.prototype.toJSON();</i> <p>JSON.stringify(value[,replacer[,space]]); Convert a JavaScript object to a JSON string.</p> <p>JSON.stringify({ a: 1, b: "2", c: true }); // -> '{"a":1,"b":"2","c":true}'</p> <p>JSON.stringify([1, 2, 3, 4, 5]); // -> "[1,2,3,4,5]"</p> <p>JSON.parse(text[,reviver]); Parses a JSON string and returns a JavaScript object.</p> <p>JSON.parse(JSON.stringify({a: 1, b: "2", c: true})); // -> Object { a: 1, b: "2", c: true }</p> <p>JSON.parse(JSON.stringify([1, 2, 3, 4, 5])); // -> Array(5) [1, 2, 3, 4, 5]</p> |

