

## JavaScript cheatsheet – v3.3.0 – <https://github.com/Serrin/Celestra/>

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

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

