

JavaScript cheatsheet – v2.2.0 – <https://github.com/Serrin/Celestra/>

Web Storage api and JSON	element.dataset & data-* attributes	TypedArray
<p>IE8 compatible</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[,length]]);</p> <p>Int8Array(); -128 to 127, 1 byte, int8_t</p> <p>Uint8Array(); 0 to 255, 1 byte, uint8_t</p> <p>Uint8ClampedArray(); 0 to 255, 1 byte, uint8_t, not in IE10-11</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>Float32Array(); 1.2x10⁻³⁸ to 3.4x10³⁸, 4 byte, float</p> <p>Float64Array(); 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>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"> - a string - a number - an object (containing valid JSON values) - an array - a boolean - null <p>Invalid Data Types</p> <ul style="list-style-type: none"> - a function - a date - undefined - an object with method(s) (function) <p>JSON.stringify() 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() 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>

DOMParser

IE9: XML support

IE10+IE11: XML, SVG and HTML support

```
var parser = new DOMParser();  
var doc = parser.parseFromString("sourceStr", "application/xml");  
Returns a Document, but not a SVGDocument nor a HTMLDocument.
```

```
var parser = new DOMParser();  
var doc = parser.parseFromString(sourceStr, "image/svg+xml");  
Returns a SVGDocument, which also is a Document.
```

```
var parser = new DOMParser();  
var doc = parser.parseFromString(sourceStr, "text/html");  
Returns a HTML document.
```

DOMParser sample function

```
function parseHTML (str) {  
  return Array.prototype.slice.call(  
    (new DOMParser())  
      .parseFromString(str, "text/html")  
      .childNodes[0]  
      .childNodes[1]  
      .childNodes  
  );  
}
```

```
parseHTML(  
  "<div>1<p>2</p><p>3</p></div>"  
  + "<div>4<p>5</p><p>6</p></div>"  
  + "<div>7</div>"  
  + "<p>8</p>"  
)  
=> Array(4) [ div, div, div, p ]  
Tested in IE11, Edge, Firefox and Chrome.
```