The most common DOM methods at a glance

**Reaching Elements in a Document**

- `document.getElementById('id')`: Retrieves the element with the given id as an object.
- `document.getElementsByTagName('tagname')`: Retrieves all elements with the tag name `tagname` and stores them in an array-like list.

**Reading Element Attributes, Node Values and Other Data**

- `node.getAttribute('attribute')`: Retrieves the value of the attribute with the name `attribute`.
- `node.setAttribute('attribute', 'value')`: Sets the value of the attribute with the name `attribute` to `value`.
- `node.nodeType`: Reads the type of the `node` (1 = element, 3 = text node).
- `node.nodeName`: Reads the name of the `node` (either element name or #textNode).
- `node.nodeValue`: Reads or sets the value of the `node` (the text content in the case of text nodes).

**Navigating Between Nodes**

- `node.previousSibling`: Retrieves the previous sibling node and stores it as an object.
- `node.nextSibling`: Retrieves the next sibling node and stores it as an object.
- `node.childNodes`: Retrieves all child nodes of the object and stores them in a list. Here are shortcuts for the first and last child node, named `node.firstChild` and `node.lastChild`.
- `node.parentNode`: Retrieves the node containing `node`.

**Creating New Nodes**

- `document.createElement(element)`: Creates a new element node with the name `element`. You provide the name as a string.
- `document.createTextNode(string)`: Creates a new text node with the node value of `string`.
- `newNode = node.cloneNode(bool)`: Creates `newNode` as a copy (clone) of `node`. If `bool` is `true`, the clone includes clones of all the child nodes of the original.
- `node.appendChild(newNode)`: Adds `newNode` as a new (last) child node to `node`.
- `node.insertBefore(newNode, oldNode)`: Inserts `newNode` as a new child node of `node` before `oldNode`.
- `node.removeChild(oldNode)`: Removes the child `oldNode` from `node`.
- `node.replaceChild(newNode, oldNode)`: Replaces the child node `oldNode` of `node` with `newNode`.
- `element.innerHTML`: Reads or writes the HTML content of the given element as a string—including all child nodes with their attributes and text content.

**Known browser quirks:**

- `getAttribute` and `setAttribute` are not reliable. Instead, assign the property of the element object directly: `obj.property = value`. Furthermore, some attributes are actually reserved words, so instead of `class` use `className` and instead of `for` use `HTMLfor`.
- Real DOM compliant browsers will return linebreaks as text nodes in the `childNodes` collection, make sure to either remove them or test for the `nodeType`.

Assembled by Christian Heilmann (http://wait-till-i.com), licensed Creative Commons Attribution (http://creativecommons.org/licenses/by/3.0/). Enjoy.