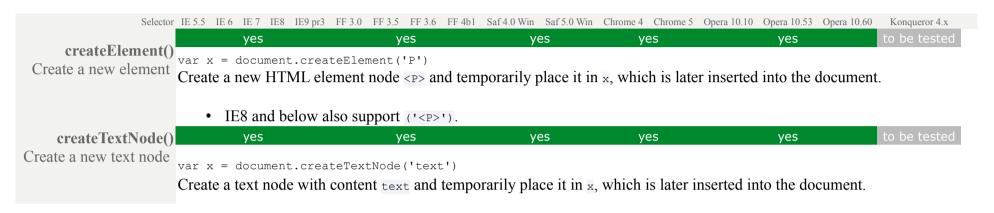
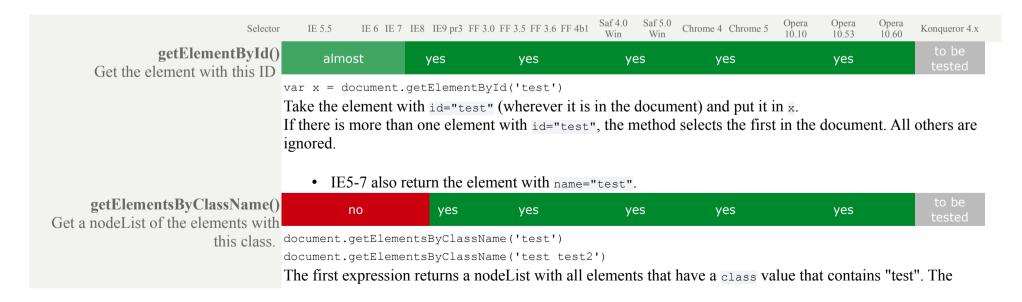
# Creating elements

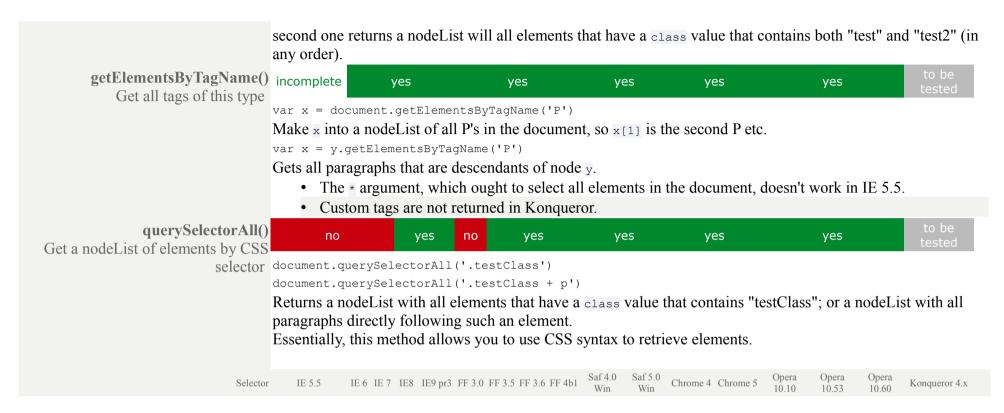
These two methods create new HTML elements which you can then insert into the document.



# Getting elements

These methods are meant for getting the HTML elements you need from the document.

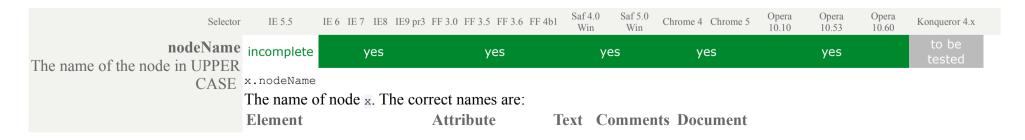


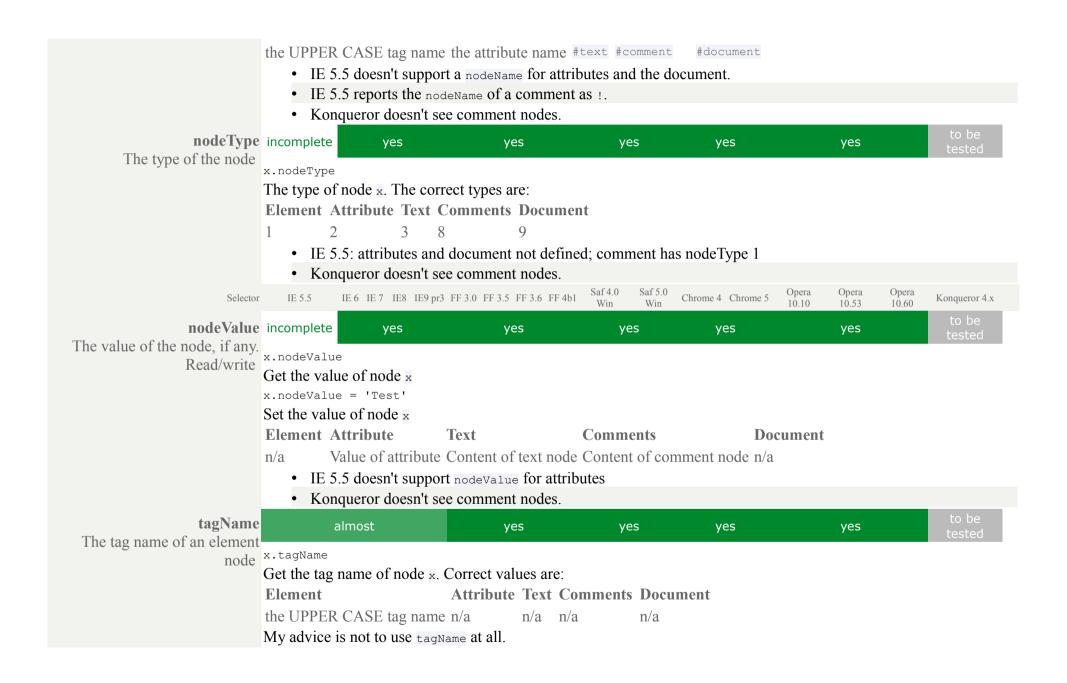


## Node information

These four properties give basic information about all nodes. What they return depends on the node type. They are read-only, except for node Value.

There are three basic node types: element nodes (HTML tags), attribute nodes and text nodes. I test these properties for all these three types and added a fourth node type: the document node (the root of all other nodes).





• In IE (all versions) the tagName of a comment node is !

Selector IE 5.5 IE 6 IE 7 IE8 IE9 pr3 FF 3.0 FF 3.5 FF 3.6 FF 4b1 Saf 4.0 Win Win Chrome 4 Chrome 5 Opera Opera 10.10 Opera 10.53 Opera 10.60 Konqueror 4.x

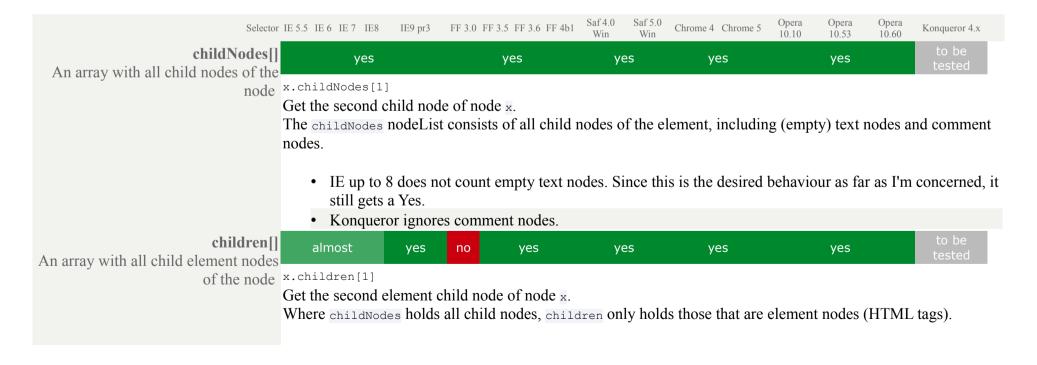
## The DOM tree

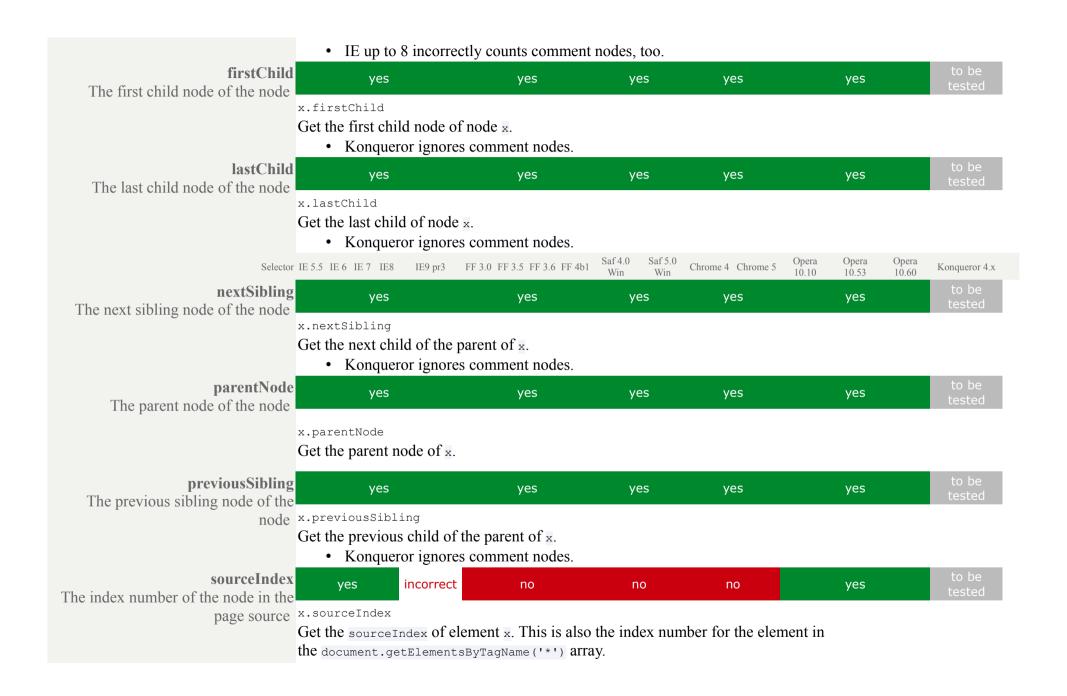
Five properties and two arrays for walking through the DOM tree. Using these properties, you can reach nodes that are close to the current node in the document structure.

In general you shouldn't use too many of these properties. As soon as you're doing something like

x.parentNode.firstChild.nextSibling.children[2]

your code is too complicated. Complex relationships between nodes can suddenly and unexpectedly change when you alter the document structure, and altering the document structure is the point of the W3C DOM. In general you should use only one or two of these properties per action.





• IE returns a sourceIndex that's one higher than in Opera. In IE8 and lower this refers to the correct element, but in IE9 it refers to the next element.

Selector IE 5.5 IE 6 IE 7 IE 8 IE 9 pr 3 FF 3.0 FF 3.5 FF 3.6 FF 4b1 Saf 4.0 Win Win Chrome 4 Chrome 5 Opera Opera Opera 10 10 53 10 60 Konqueror 4.x

## **DOM Traversal**

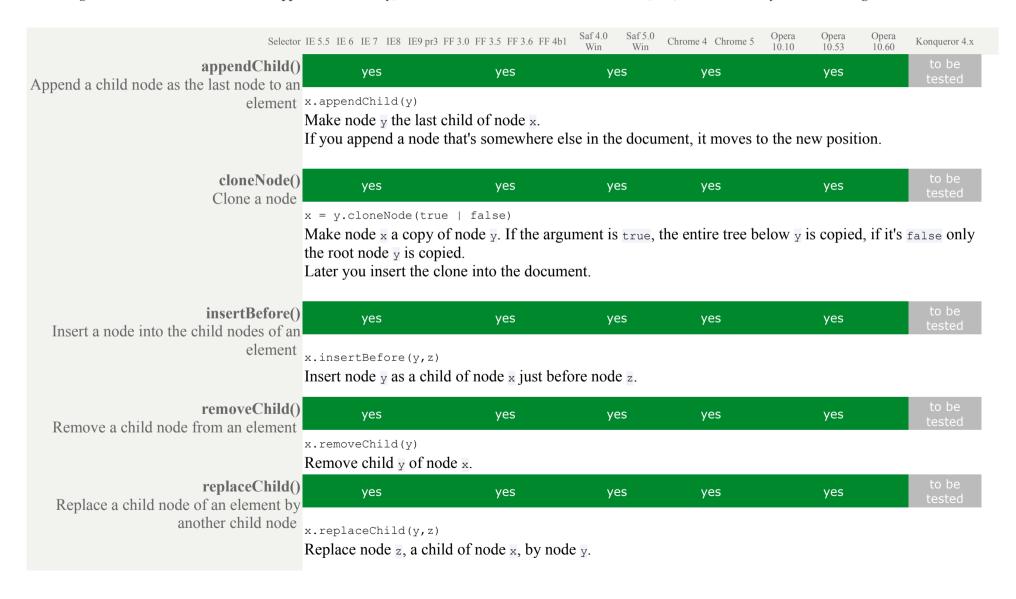
A few useful properties that should have been in the DOM from the start but mysteriously weren't.



# Node manipulation

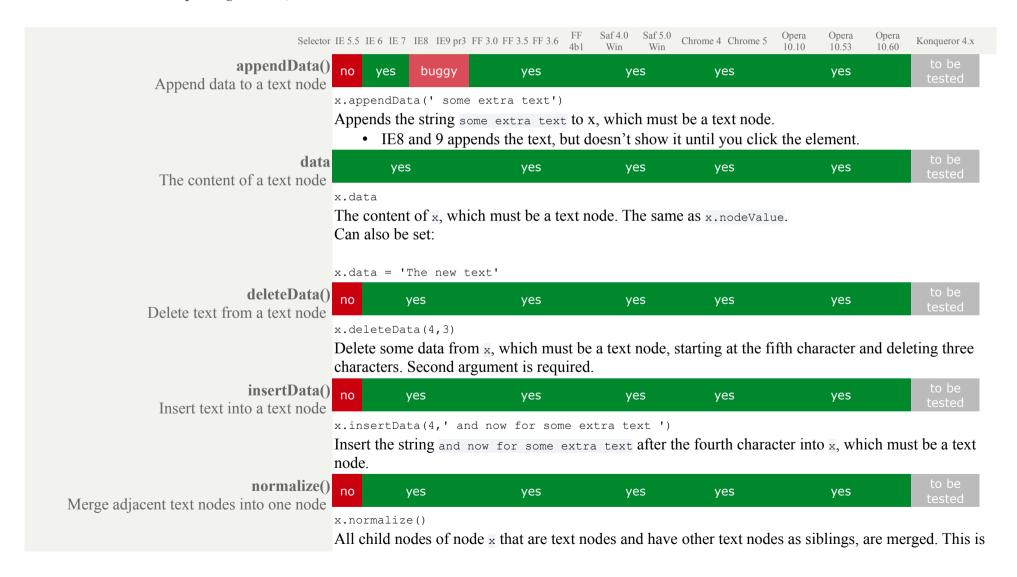
These five methods allow you to restructure the document. The average DOM script uses at least two of these methods.

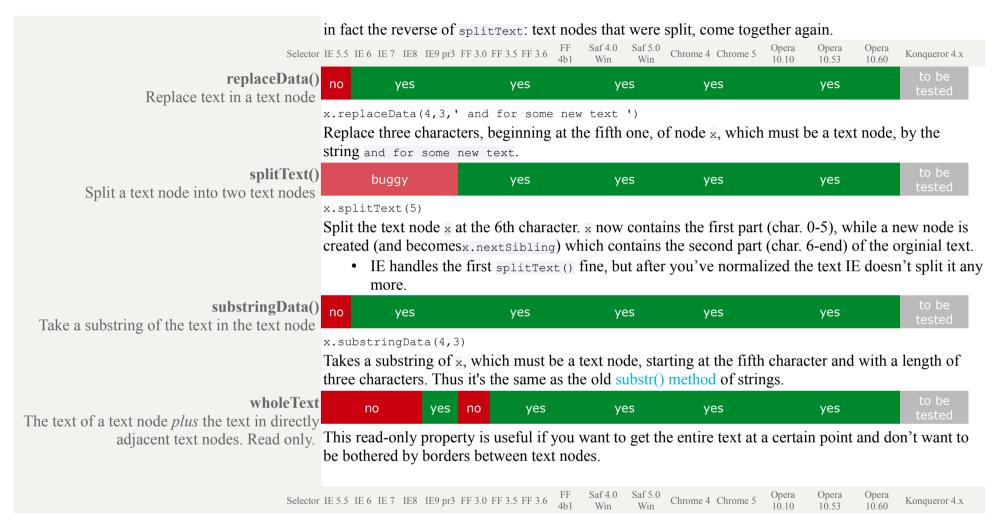
The changes in the document structure are applied immediately, the whole DOM tree is altered. The browser, too, will immediately show the changes.



#### Text data

These methods are for manipulating text data, i.e. the contents of text nodes.



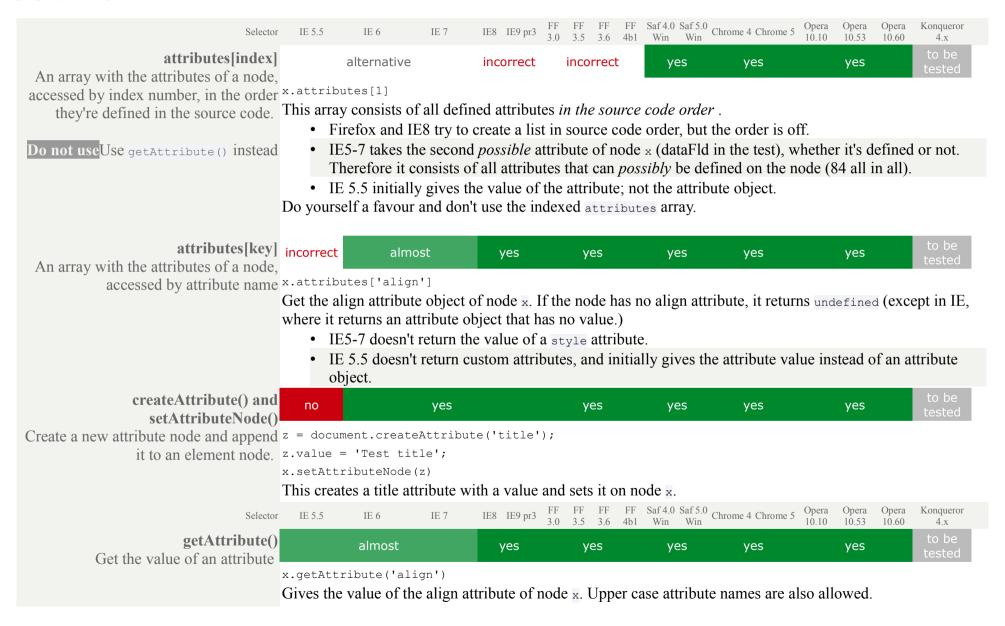


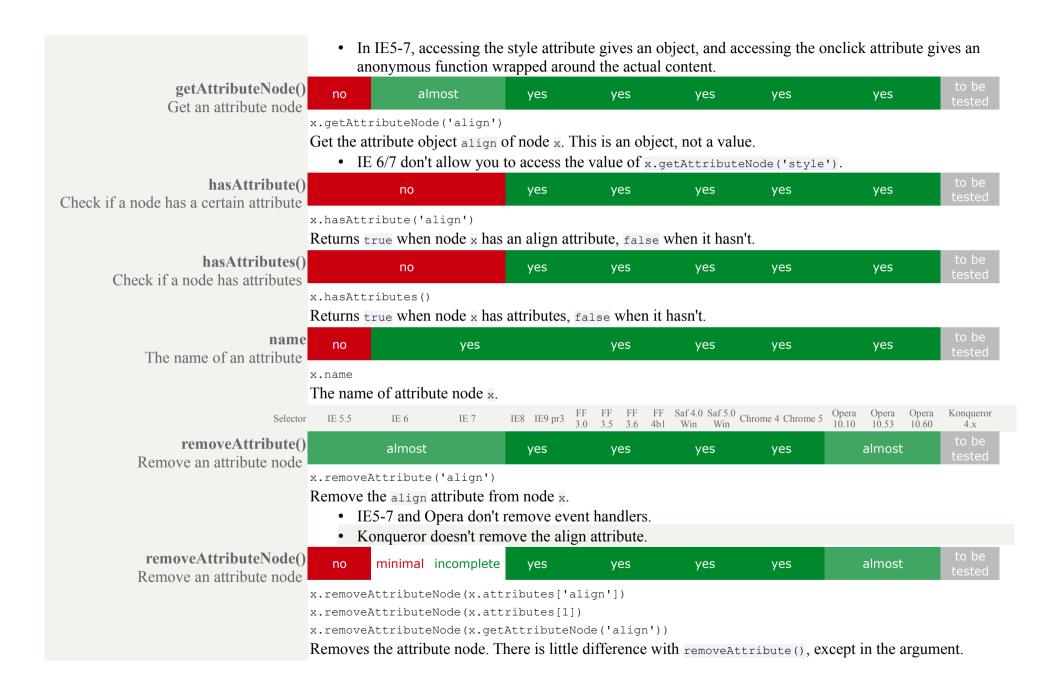
# **Attributes**

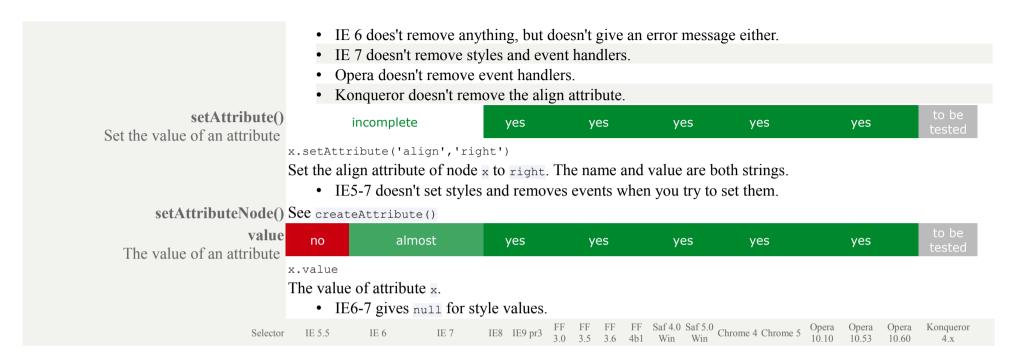
A bloody mess. Try influencing attributes in this order:

- 1. Try getting or setting a specific property, like x.id or y.onclick.
- 2. If there is no specific property, use getAttribute() orsetAttribute().
- 3. If even that doesn't work, try any other method or property in the table below. Most have horrible browser incompatibility patterns, though.
- 4. Avoid attributes[]. It's worse than anything else.

In my view any method or property concerning attribute nodes should also work on the style attribute, event handlers and custom attributes. If not I judge the method or property incomplete.

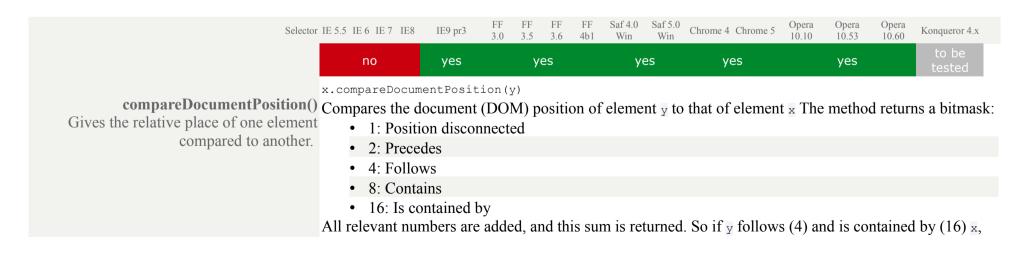


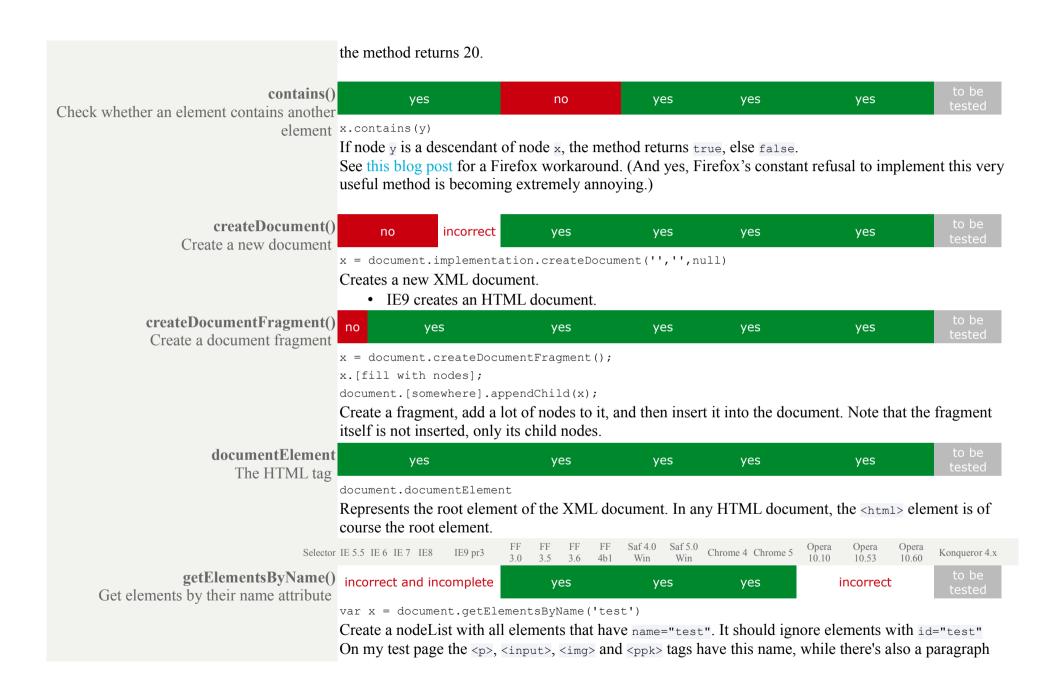


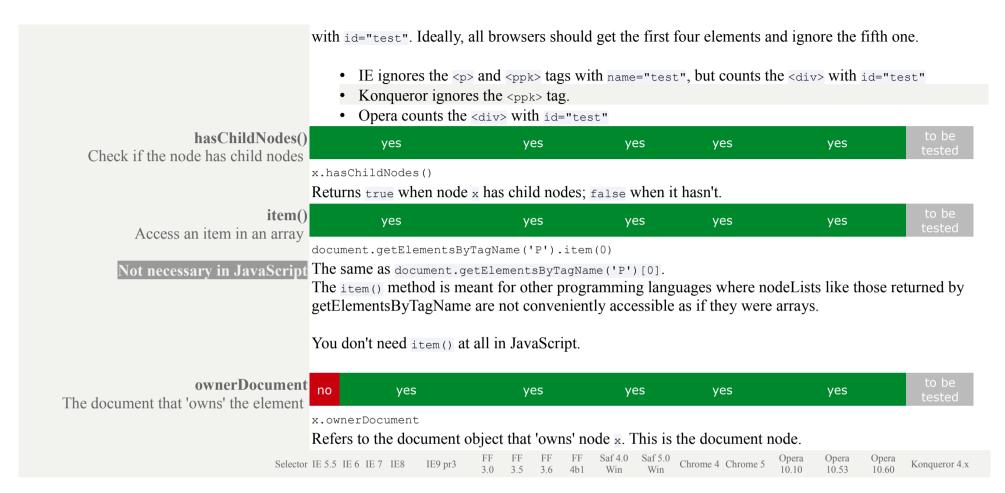


# Miscellaneous

A lot of miscellaneous methods and properties that you'll rarely need. I use only two of them in an actual script.







## Microsoft extensions

As usual Microsoft has extended the standard somewhat. Though sometimes its extensions are brilliant (innerHTML springs to mind), in the case of the DOM Core they aren't.

Note the difference between W3C and Microsoft methods. The W3C methods are owned by the parent element of the node you want to adjust, the Microsoft methods by the node itself.

