Keep in Mind

* To quickly identify an element use magnifying glass
* Assign a node from DOM to variable for further use
* Everything in the html is the node including text

Compatibility Check

* caniuse.com is a great resource for checking compatibility i.e. methods or property is compatible with browser or not.
* W3C DOM Compatibility – www.quirksmode.org
* HTML5 Cross Browser Polyfills – goo.gl/LPTDP
* Pollyfills, Shims and More – www.calormen.com/polyfill

Selecting DOM Elements

Choosing by Id

* getElementById() most common
* Tags with a specific ID
* A single ID per page
* Most common error is people often type getElementsById()
* Instead of use console.log its more better to use console.dir to explore the Node all properties and available methods

Choosing by Tag

* Traditionally most popular way to access an element by using getElementById() the next most popular one is getElementsByTagName()
* getElementsByTagName()
* Groups elements by tags
* Returns an array
* Can be combined with getElementById()
* getElementsByTagName is super handy specially when we combine the id with tag.

Elements by Class Name

* getElementsByClassName()
* Elements with a specific class
* Newer selector
* Not compatible with older browsers

Querying with CSS Selectors

* Selecting elements in JavaScript little bit harder using getElementById(), getElementsByTagName(), getElementsByClassName() in newer version of the browser
* One of the best way to isolating or selecting elements by querySelector(), querySelectorAll()
* Easiest way to selecting the elements
* There is a difference between querySelector() and querySelectorAll() is that queySelector() return single element whereas querySelectorAll() returns an array.
* Nodes through CSS selectors
* Similar to jQuery
* Not compatible with older browsers

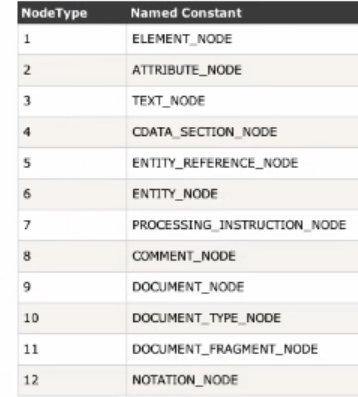
Selecting named form elements

* Form elements can have name attributes
* DOM provides document.forms object
* Named elements can also be selected

Node Properties

* node.nodeType – numerical value of a node
* node.nodeName – the name of the node
* node.nodeAttributes – array of node attributes
* node.nodeValue – element inside a node – stores the value of the node (null for element nodes and the text for text nodes)
* We have to remember technically everything in the dom is the node including text elements
* If we have to change the text of a node first include firstChild then use nodeValue

Node Types



* We generally used element, attribute and text node.

Traversing up and down DOM Nodes

Moving up and down

* parentNode – Goes up a level
* childNodes – Array of children
* firstChild/lastChild – First/Last element
* previousSibling/nextSibling - Elements with same parent
* Navigating node in this way may be easy it has some costs like sometimes return carriage return

Targeting Node Elements

* firstElementChild – It will return First element child only if it an element like tag it will ignore comment or text.
* lastElementChild – Last element child
* children – Only children that are elements
* previousElementSibling/nextElementSibling
* This has also lousy browser support