Survey of HTML

Relevant Links

- Flanagan's book, Chapter 15
- MDN's DOM documentation[1]
- Eloquent Javascript[2]
- MDN's page on document[3]
- MDN's page on elements[4]
- devdocs[5]

Notes

The **Document Object Model** is an API for accessing HTML and XML documents.

It represents the document as a tree, the *root node* being the html/xml tag.

Each node has:

children all the nodes that are immediately below the node.
siblings children of the same node are siblings to each other.
descendents all the nodes that are somewhere below the node (so all their children, their children's children, their children's children's children and so on).
parent Each non-root node has a parent node.
ancestors The node's parent, its parent's parent, its parent's parent's parent and so on.

DOM Nodes comes in various types. Even though you might expect that only tag elements would be nodes, in fact this is in practice not the case: There are *text nodes*, *attribute nodes*, *comment nodes* and so on. Some of the DOM methods bypass non-element nodes. Also many popular libraries, like jQuery, offer a more convenient interface.

The DOM interface also allows you to manipulate the page elements, not only access them. We will see examples in the future.

Here is a main list of the properties and methods you have access to. Most of these apply to both the document object and any specific element objects.

getElementById[6] Returns the element node in the document with a given id.
getElementsByTagName[7] Returns all elements with a specific tag.
getElementsByClassName[8] Returns all elements with a specific class.

[1] https://developer.mozilla.org/fi/docs/DOM
[2] http://eloquentjavascript.net/13_dom.html
[3] https://developer.mozilla.org/en-US/docs/Web/API/Document
[4] https://developer.mozilla.org/en-US/docs/Web/API/Element
[5] http://devdocs.io/

querySelector[9] Returns the first element that matches the given selector.
querySelectorAll[10] Returns all elements that match the given selector.
children[11] Returns all the element nodes that are children of the current node.
firstElementChild[12] Returns the first element node that is a child of the current node.
innerHTML[13] A property specifying the contents of the node (as a string). Also see textContent[14]
outerHTML[15] A property specifying the entire node with its contents (as a string).
className[16] The string containing the element's classes (space-separated).
tagName[17] The element's tag name.
id[18] The element's id.
getAttribute[19]**, setAttribute**[20] Get and set the attribute with a specific name.

We will not spend more time with this part of the DOM, and we will instead rely on jQuery for DOM manipulation. But you should be aware of what is out there, and that you could do directly most of the things that jQuery offers.

Chapter 15 of Flanagan's book, along with sections of Part IV, do a very good job showing what is possible as well as how one would manipulate a page with the standard DOM toolkit.

Before we move on however, let us examine a small example, using the current page as a test:

```
let dls = document.getElementsByTagName("dl");     // definition lists
let terms = dls[0];                 // the definition list near the top
for (const child of terms.children) {
  if (child.tagName == "DT") {
    console.log(child.textContent);
  }
}
let newElement = document.createElement("dt");
newElement.textContent = "hi!";
terms.appendChild(newElement);
let newElement2 = document.createElement("dd");
newElement2.textContent = "Definition of 'hi'";
terms.appendChild(newElement2);
```

[14] https://developer.mozilla.org/en-US/docs/Web/API/Node.textContent