30th November 2019

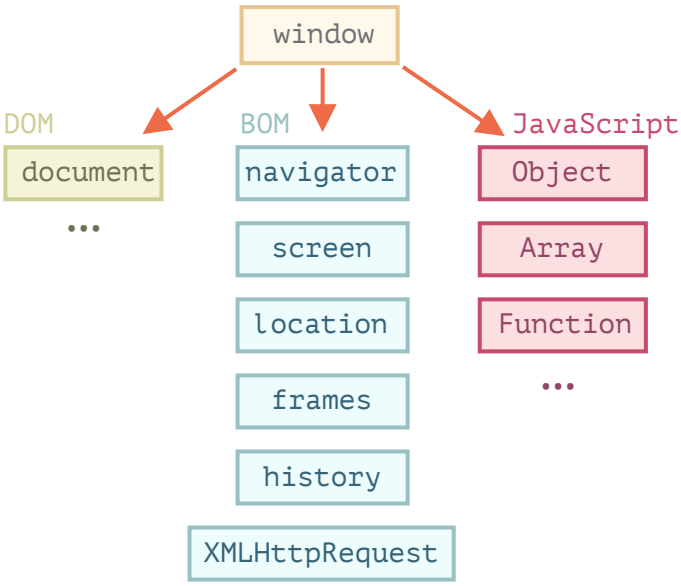
Browser environment, specs

The JavaScript language was initially created for web browsers. Since then it has evolved and become a language with many uses and platforms.

A platform may be a browser, or a web-server or another *host*, even a coffee machine. Each of them provides platform-specific functionality. The JavaScript specification calls that a *host environment*.

A host environment provides own objects and functions additional to the language core. Web browsers give a means to control web pages. Node.js provides server-side features, and so on.

Here's a bird's-eye view of what we have when JavaScript runs in a web-browser:



There's a "root" object called `window`. It has two roles:

1. First, it is a global object for JavaScript code, as described in the chapter Global object.
2. Second, it represents the "browser window" and provides methods to control it.

For instance, here we use it as a global object:

```
1  function sayHi() {
2    alert("Hello");
3  }
4
5  // global functions are methods of the global object:
6  window.sayHi();
```

And here we use it as a browser window, to see the window height:

```
1  alert(window.innerHeight); // inner window height
```

There are more window-specific methods and properties, we'll cover them later.

DOM (Document Object Model)

Document Object Model, or DOM for short, represents all page content as objects that can be modified.

The `document` object is the main "entry point" to the page. We can change or create anything on the page using it.

For instance:

```
1  // change the background color to red
2  document.body.style.background = "red";
3
4  // change it back after 1 second
5  setTimeout(() => document.body.style.background = "", 1000);
```

Here we used `document.body.style` , but there's much, much more. Properties and methods are described in the specification:

- **DOM Living Standard** at https://dom.spec.whatwg.org

DOM is not only for browsers

The DOM specification explains the structure of a document and provides objects to manipulate it. There are non-browser instruments that use DOM too.

For instance, server-side scripts that download HTML pages and process them can also use DOM. They may support only a part of the specification though.

CSSOM for styling

CSS rules and stylesheets are structured in a different way than HTML. There's a separate specification, CSS Object Model (CSSOM), that explains how they are represented as objects, and how to read and write them.

CSSOM is used together with DOM when we modify style rules for the document. In practice though, CSSOM is rarely required, because usually CSS rules are static. We rarely need to add/remove CSS rules from JavaScript, but that's also possible.

BOM (Browser Object Model)

The Browser Object Model (BOM) represents additional objects provided by the browser (host environment) for working with everything except the document.

For instance:

- The navigator object provides background information about the browser and the operating system. There are many properties, but the two most widely known are: `navigator.userAgent` – about the current browser, and `navigator.platform` – about the platform (can help to differ between Windows/Linux/Mac etc).
- The location object allows us to read the current URL and can redirect the browser to a new one.

Here's how we can use the `location` object:

```
1 alert(location.href); // shows current URL
2 if (confirm("Go to Wikipedia?")) {
3   location.href = "https://wikipedia.org"; // redirect the browser to another
4 }
```

Functions `alert/confirm/prompt` are also a part of BOM: they are directly not related to the document, but represent pure browser methods of communicating with the user.

Specifications

BOM is the part of the general HTML specification.

Yes, you heard that right. The HTML spec at https://html.spec.whatwg.org is not only about the "HTML language" (tags, attributes), but also covers a bunch of objects, methods and browser-specific DOM extensions. That's "HTML in broad terms". Also, some parts have additional specs listed at https://spec.whatwg.org.

Summary

Talking about standards, we have:

DOM specification

Describes the document structure, manipulations and events, see https://dom.spec.whatwg.org.

CSSOM specification

Describes stylesheets and style rules, manipulations with them and their binding to documents, see https://www.w3.org/TR/cssom-1/.

HTML specification

Describes the HTML language (e.g. tags) and also the BOM (browser object model) – various browser functions: `setTimeout`, `alert`, `location` and so on, see https://html.spec.whatwg.org. It takes the DOM specification and extends it with many additional properties and methods.

Additionally, some classes are described separately at https://spec.whatwg.org/.

Please note these links, as there's so much stuff to learn it's impossible to cover and remember everything.

When you'd like to read about a property or a method, the Mozilla manual at https://developer.mozilla.org/en-US/search is also a nice resource, but the corresponding spec may be better: it's more complex and longer to read, but will make your fundamental knowledge sound and complete.

To find something, it's often convenient to use an internet search "WHATWG [term]" or "MDN [term]", e.g https://google.com?q=whatwg+localstorage, https://google.com?q=mdn+localstorage.

Now we'll get down to learning DOM, because the document plays the central role in the UI.

Previous lesson Next lesson

Share

Tutorial map

Comments

- If you have suggestions what to improve - please submit a GitHub issue or a pull request instead of commenting.
- If you can't understand something in the article – please elaborate.
- To insert a few words of code, use the `<code>` tag, for several lines – use `<pre>`, for more than 10 lines – use a sandbox (plnkr, JSBin, codepen…)