Now before we dive into DOM, let’s talk about web browsers.

When you try to open a webpage, you go to a browser's address bar, typing [www.google.com](http://www.google.com). You hit enter and you wait, because something must happen and it does, a webpage with a picture of google and a few elements appears right away. Now how did this happen exactly? Have you ever thought about it?

Imagine the following process. This here is the client and this is the server. The client is a program, which in our case is the browser. The server is just a server. Now first, when you type a URL into the browser’s address bar and you hit enter, the browser will go and ask one thing, named DNS, where is google.com? Without going into details, the DNS will look it up and say, google.com is located on this IP address. From there, knowing the exact IP address, where google.com is being hosted, our browser will send an HTTP request towards that server over the internet. HTTP stands for Hypertext Transfer Protocol, which is a network protocol for retrieving named resources, like chunks of information, such as web pages or pictures.

On the other side there is the server, which can have many things, services and programs on specific port numbers. However, when an HTTP request arrives from the internet, the server passes it to port 80, which is the webserver. The webserver is just program, which listens for requests and answers them. In our case the webserver accepts an HTTP request, which is in the following format

GET www.google.com HTTP/1.1

and returns an appropriate HTTP response, in plain text. Now this response travels back to your IP address and goes back to your client, the browser. Without going into details about what Hyper Text is, we can say that your browser receives plain text. That’s right, it does not receive a website, it receives a text. And this is where it gets interesting, the browser must understand what this text is and it does so by parsing it. The HTTP protocol contains headers, which the browser uses in order to properly parse the response and ultimately build a tree of elements, which we call the Document Object Model.

Basically, whenever you try to open a web site, the browser sends a request and receives a response, which he parses, building a tree. Why is it a tree, but not a graph? Well because the nodes of every document are hierarchically organized in a tree structure, with the topmost node named as "Document Object".