The Document Object Model (DOM) represents an HTML document as a tree structure, where each node is an object representing a part of the document. Here's a brief overview of the DOM tree structure and how JavaScript interacts with it:

**DOM Tree Structure**

The DOM tree starts with the document object, which is the root of the tree. Below it, we have the html element, which contains the head and body elements. Each of these elements can have child elements, forming a hierarchical tree structure.

For example, consider this simple HTML document:

<!DOCTYPE html>

<html>

<head>

<title>Sample Page</title>

</head>

<body>

<h1>Hello, World!</h1>

<p>This is a sample page.</p>

</body>

</html>

The DOM tree for this document looks like this:

document

└── html

├── head

│ └── title

│ └── #text (Sample Page)

└── body

1. ├── h1
2. │ └── #text (Hello, World!)
3. └── p
4. └── #text (This is a sample page.)

**JavaScript Interaction with the DOM**

JavaScript can interact with the DOM to manipulate the structure, style, and content of web pages. The placement of JavaScript in the HTML document affects how and when it can interact with the DOM.

**JavaScript in the <head> Section**

When JavaScript is placed in the <head> section, it is executed before the DOM is fully loaded. This can cause issues if the script tries to access elements that haven't been created yet. To avoid this, you can use the defer attribute in the <script> tag, which tells the browser to execute the script after the DOM has been fully parsed.

<!DOCTYPE html>

<html>

<head>

<title>Sample Page</title>

<script src="script.js" defer></script>

</head>

<body>

<h1>Hello, World!</h1>

<p>This is a sample page.</p>

</body>

</html>

**JavaScript in the <body> Section**

Placing JavaScript at the end of the <body> section ensures that the DOM is fully loaded before the script runs. This is a common practice to avoid issues with accessing DOM elements.

<!DOCTYPE html>

<html>

<head>

<title>Sample Page</title>

</head>

<body>

<h1>Hello, World!</h1>

<p>This is a sample page.</p>

<script src="script.js"></script>

</body>

</html>

**Example of JavaScript Manipulating the DOM**

Here's an example of JavaScript code that changes the text of the <h1> element:

document.addEventListener("DOMContentLoaded", function() {

document.querySelector("h1").textContent = "Hello, JavaScript!";

});

This script waits for the DOM to be fully loaded before changing the text content of the <h1> element.

Understanding the DOM and how JavaScript interacts with it is fundamental for creating dynamic and interactive web pages[[1]](https://www.freecodecamp.org/news/javascript-in-the-browser-dom-and-events/)[[2]](https://javascript.info/dom-nodes)[[3]](https://dev.to/gautam_kumar_d3daad738680/understanding-the-dom-tree-a-beginners-guide-to-understanding-web-page-structure-2ncd). If you have any more questions or need further clarification, feel free to ask!

**We can add element dynamically**

**We can change content**

**We can remove from page**