# write a blog on differences between document object and windows object in JavaScript

In the realm of JavaScript and web development, understanding the Document Object Model (DOM) is fundamental. Two crucial components of the DOM are the Document object and the Window object. Let's delve into their disparities in a tabular format, showcasing the objects nested within each:

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Document Object** | **Window Object** |
| **Scope** | Limited to the content of the web page. | Represents the entire browser window. |
| **Hierarchy** | Nested within the Window object. | Contains the Document object and other window-related properties. |
| **Accessing Elements** | Provides methods like **getElementById**, **getElementsByClassName**, **getElementsByTagName**. | Provides properties like **document**, **parent**, and **frames** to access the Document object. |
| **Events** | Handles events related to document structure (e.g., clicks on elements, form changes). | Handles events related to the browser window (e.g., resizing, closing). |
| **Lifecycle** | Tied to the loading and unloading of the web page. Represents static content. | Tied to the browser window's lifecycle, persists across page loads within the window. |

### **Objects Within Each:**

Let's explore the objects nested within both the Document and Window objects:

#### **Objects Within Document Object:**

1. **Element**: Represents an HTML element and provides methods and properties to manipulate it.

var myElement = document.getElementByID(“myElement”);

1. **NodeList**: Represents a collection of nodes (e.g., elements) and provides methods to traverse and manipulate them.

**var nodeList = document.**getElementByTagName(“p”);

1. **Form**: Represents an HTML form element and provides methods and properties to interact with it.

**var myForm = document.**forms[“myForms”];

#### **Objects Within Window Object:**

1. **Navigator**: Provides information about the browser.

**var browserInfo = window**.navigator.userAgent;

1. **History**: Represents the browsing history of the window.

**Var historyLength = window.**history.length;

1. **Location**: Represents the URL of the window and provides methods to navigate to different URLs.

**Var currentURL = window**.location.href;

### **Conclusion:**

Understanding the Document object and the Window object, along with the objects nested within them, is crucial for proficient JavaScript development. By grasping their distinctions and knowing how to utilize each object effectively, developers can build dynamic and interactive web applications. Whether it's manipulating document elements or controlling the browser window, a solid understanding of these concepts is essential for success in web development.