**Differences between Document and Window Objects**

In JavaScript, the document object and the window object are both important components of the Document Object Model (DOM) and serve different purposes.

**Document Object:**

The document object represents the web page loaded in the current window or frame. It provides access to the content and structure of the document. Some key features and properties of the document object include:

* DOM manipulation: The document object provides methods like getElementById, createElement, and appendChild to manipulate the elements within the document.
* Access to elements: It allows you to access and modify HTML elements, their attributes, and content.
* Event handling: You can attach event listeners to elements using methods like addEventListener to respond to user interactions.
* Forms and input handling: The document object provides methods to access and manipulate form elements and their values.

**Example:**

// Accessing an element by its ID

const element = document.getElementById('myElement');

// Creating a new element

const newElement = document.createElement('div');

// Appending an element to a parent element

document.body.appendChild(newElement);

// Adding an event listener

element.addEventListener('click', () => {

// Handle click event

});

* DOM Tree: The document object represents the root of the Document Object Model (DOM) tree, which is a hierarchical representation of the HTML structure of a web page. It allows you to traverse and manipulate the elements within the document.
* Methods for element selection: The document object provides various methods for selecting elements within the document, such as getElementById, getElementsByClassName, getElementsByTagName, and querySelector. These methods return references to one or more elements that match the given selector.
* Content manipulation: You can modify the content of elements using properties like innerHTML, which allows you to set or retrieve the HTML content within an element. The textContent property can be used to set or retrieve the text content, excluding any HTML tags.
* Styling and CSS: The document object provides methods to dynamically modify the styles and classes of elements. For example, element.style allows you to directly manipulate the inline styles of an element, and the classList property provides methods to add, remove, or toggle CSS classes.
* Form handling: The document object allows interaction with HTML forms. You can access form elements, retrieve or set their values, and listen for form submission events using the submit event.

**window Object:**

The window object represents the current browser window or tab. It provides access to various properties and methods related to the browser window. Some key features and properties of the window object include:

* Global scope: Variables and functions defined in the global scope are accessible through the window object.
* Navigation and location: The window object provides methods like open, close, and properties like location to control the browser window's navigation and URL.
* Window size and position: You can resize and reposition the browser window using methods like resizeTo and moveTo.

\*Timers and intervals: The window object provides methods like setTimeout and setInterval to execute code at specified time intervals.

**Example:**

// Opening a new window

window.open('https://www.example.com');

// Closing the current window

window.close();

// Resizing the window

window.resizeTo(800, 600);

// Accessing the current URL

const currentURL = window.location.href;

// Executing code after a delay

setTimeout(() => {

// Code to be executed after a delay

}, 2000);

* Global scope and variables: In a browser environment, variables and functions declared in the global scope are attached to the window object. For example, if you define a global variable x, you can access it as window.x.
* Window properties: The window object provides access to various properties related to the browser window, such as window.innerWidth and window.innerHeight, which give the dimensions of the viewport. The window.location property provides information about the current URL and allows navigation to a new URL.
* Window methods: The window object provides several methods for controlling the browser window. For instance, you can open a new browser window or tab using window.open(), close the current window using window.close(), or navigate to a new URL using window.location.href.
* Timers and intervals: The window object includes methods for executing code at specified intervals. setTimeout() allows you to execute a function after a specified delay, while setInterval() repeatedly executes a function at a specified interval until cleared with clearInterval().
* Browser events: The window object allows you to listen for and handle various browser events such as resize, scroll, load, and unload. These events provide hooks to execute code in response to specific actions or changes in the browser window.

Both the document object and the window object provide powerful capabilities for interacting with and manipulating web pages. They are fundamental components of client-side JavaScript and are extensively used together to create dynamic and interactive web applications.

In summary, the document object is used to manipulate the content and structure of the loaded document, while the window object provides access to browser window-related properties and methods. Although they are separate objects, they are closely related and often used together to create dynamic and interactive web pages.