1.write a blog on the difference between document & window objects**Document** Object:

The `document` object is at the heart of every web page. It represents the entire HTML content of the page and provides an interface for developers to manipulate that content. Essentially, the `document` object is a representation of the structured content that makes up the page, including all its HTML elements, text, images, and other assets.

Here are some key characteristics of the `document` object:

1. **Hierarchical Structure**: The `document` object represents the entire DOM tree. It's structured hierarchically, with each HTML element being a node in the tree. This structure allows developers to traverse, locate, and manipulate specific elements.
2. **Methods for Element Access**: The `document` object provides methods to access elements within the DOM using various strategies, such as by their ID, class name, tag name, or CSS selector. For

instance, `getElementById()`, `getElementsByClassName()`, and `querySelector()` are methods commonly used for element retrieval.

3. **Content Manipulation**: Through the `document` object, developers can change the content, attributes, and styles of elements on the page. This enables dynamic updates to reflect user interactions and real-time data changes.

Window Object:

While the `document` object focuses on the content of the web page, the `window` object is concerned with the broader environment in which the page is displayed. It represents the browser window or tab that contains the web page and provides methods and properties for managing the window itself and interacting with the user.

Here are some key characteristics of the `window` object:

1. **Global Scope**: The `window` object is the highest-level object in the browser's JavaScript environment. All global variables and functions are properties of the `window` object. This means that you can access global variables directly as properties of `window`.
2. **Browser Controls**: The `window` object provides methods to control the browser, such as `open()` for opening new browser windows or tabs and `close()` for closing them. It also handles navigation functions like `history.back()` and `history.forward()` to move backward and forward through the user's browsing history.
3. **Timers and Intervals**: The `window` object allows you to create timers and intervals using `setTimeout()` and `setInterval()`, enabling you to execute code at specified intervals.
4. **Interaction with User**: The `window` object handles user interactions like displaying alerts, confirmations, and prompts through methods like `alert()`, `confirm()`, and `prompt()`.