The Document Object Model (DOM) is a programming interface for HTML and XML documents. It represents the structure of a document as a tree-like hierarchy, with the document itself at the top and individual elements, such as headings and paragraphs, nested inside. The DOM allows developers to access and manipulate the content and structure of a document using JavaScript.

The Window object represents the browser window and is the global object in the browser. It provides access to the browser's functionality, such as the ability to open and close new windows, as well as access to the Document object, which represents the current web page.

In short, the Document object is used to access and manipulate the content and structure of a web page, while the Window object is used to access the browser's functionality and the Document object.

In addition to Document and Window, JavaScript also has a number of other built-in objects, such as the Navigator and Location objects, which provide information about the browser and the current web page, respectively.

To sum up,

* The Document object is used to represent the content and structure of a web page.
* The Window object is used to represent the browser window and provide access to the browser's functionality.
* It is important to note that the Document object is a property of the Window object, so you can access it through the window. document property

The Document object has several properties that can be used to access different parts of the web page, such as:

* **document.documentElement**: Returns the root element of the document (usually the <html> element).
* **document. head**: Returns the <head> element of the document.
* **document.body**: Returns the <body> element of the document.
* document.title: Returns the title of the document. document.URL: Returns the full URL of the document.
* document.domain: Returns the domain name of the document's URL.
* document.referrer: Returns the URL of the document that loaded the current document. document.images: Returns a collection of all <img> elements in the document. document.scripts: Returns a collection of all <script> elements in the document. document.forms: Returns a collection of all <form> elements in the document.

The Window object also has several properties that can be used to access different parts of the browser, such as:

* window.innerWidth: Returns the width of the browser window's viewport (the part of the web page that is visible on the screen).
* window.innerHeight: Returns the height of the browser window's viewport.
* window.screenX: Returns the horizontal position of the browser window on the screen. window.screenY: Returns the vertical position of the browser window on the screen. window.location: Returns the Location object, which can be used to get information about the current web page's URL.
* window.navigator: Returns the Navigator object, which can be used to get information about the browser, such as the browser name and version.
* window.history: Returns the History object, which can be used to navigate through the browser's history and manipulate the browser's session history. It's worth noting that some of these properties and methods are now considered as outdated and modern javascript has replaced them with other alternatives.

In modern JavaScript, some of the properties and methods that have been replaced by more updated alternatives include:

* + - Instead of using document.images and document.scripts, you can use document.querySelectorAll("img") and document.querySelectorAll("script") respectively,
    - to select all elements of a specific type on a page. Instead of using document.forms, you can use document.querySelectorAll("form"),
    - to select all forms on a page. Instead of using window.innerWidth and window.innerHeight,
    - you can use window.innerWidth and window.innerHeight respectively, to get the width and height of the viewport.
    - Instead of using window.screenX and window.screenY, you can use window.screenX and window.screenY respectively,
    - to get the horizontal and vertical position of the window on the screen. In addition, modern JavaScript also has new properties and methods like window.matchMedia(), that can be used to test if a media query is true or false, window.scrollX and window.scrollY
    - to get the current x and y scroll position of the page, window.requestAnimationFrame() for performace-efficient animations and many more. It's worth noting that the modern properties and methods are still subject to browser compatibility, so you should always check the compatibility before using them

|  |
| --- |
| // Get the title of the document  console.log(document. title);  // Get the URL of the document  console.log(document.URL);  // Get the domain name of the document's URL  console.log(document. domain);  // Get the referrer of the current document  console.log(document. referrer);  // Get the root element of the document  console.log(document.documentElement);  // Get the <head> element of the document  console.log(document. head);  // Get the <body> element of the document  console.log(document. body);  // Get all <img> elements in the document  console.log(document.querySelectorAll("img"));  // Get all <script> elements in the document  console.log(document.querySelectorAll("script"));  // Get all <form> elements in the document  console.log(document.querySelectorAll("form")); |

Here is an example of how to use some of the properties of the Window object in JavaScript:

|  |
| --- |
| // Get the current width and height of the viewport  console.log(window.innerWidth);  console.log(window.innerHeight);  // Get the current horizontal and vertical position of the window on the screen  console.log(window.screenX);  console.log(window.screenY);  // Get the current x and y scroll position of the page  console.log(window.scrollX);  console.log(window.scrollY);  // Open a new page  window.open("https://www.example.com", "\_blank");  // Check if a media query is true or false  console.log(window.matchMedia("(prefers-color-scheme: dark)").matches);  // Request Animation frame for performance-efficient animations  const animate = () => {  // animation code  window.requestAnimationFrame(animate);  }  animate(); |