Difference Between Document and Window Object

**Document Object**: The document object represent a web page that is loaded in the browser. By accessing the document object, we can access the element in the HTML page. With the help of document objects, we can add dynamic content to our web page. The document object can be accessed with a window.document or just document

**Syntax**: document.property\_name;

Properties of document:

* **activeElement**: It returns the currently active elements in the document.
* **body**: It returns the contents of the body element.
* **anchors**: It returns all <a> elements that have a name attribute.
* **baseURI**: It returns a string value that represents the base URI of the document.
* **cookie**: It returns the cookie of the current document.
* **charSet**: It returns a string, representing the document’s character encoding.
* **defaultView**: It returns the current Window Object.
* **designMode**: It is used to set documents as editable or read-only.
* **domain**: It returns the domain name of the document server.
* **doctype**: It returns the document’s doctype.
* **embeds**: It returns the collection of all embedded elements.
* **URL**: It returns the complete URL of the document.
* **forms**: It returns all the elements of the form.
* **fullScreenElement**: It returns the element that is currently present in full-screen mode.
* **title**: It returns the title element of the document.
* **head**: It returns the head element of the document.
* **links**: It returns all <area> and <a> elements that have a href attribute.
* **lastModified**: It returns the date and time of the current document that was last modified.
* **images**: It returns the collection of <img> elements in the document.
* **implementation**: It returns the DOMImplementation object associated with the current document.
* **readyState**: It returns the loading status of the current document.
* **referrer**: It returns the URI of the page that is linked to the current page.
* **scripts**: It returns all script elements present in the document.
* **strictErrorChecking**: It sets or returns whether strict error checking can be enforced on a document or not.

**Methods** **of** **Document**:

**Syntax**: document.method\_name;

The lists of most commonly used methods are listed below:

* **addEventListener**(): It is used to attach an event handler to the specified element.
* **adoptNode**(): It is used to adopt a node from another document and it returns a node object, representing the adopted node.
* **close**(): It is used to close the output stream.
* **createAttribute**(): It is used to create an attribute node with the specified name and returns the attribute object.
* **createComment**(): It is used to create a comment node with some text.
* **createDocumentFragment**(): It is used to create the document fragment to change the content of the document.
* **createElement**(): It is used to create HTML element .
* **createEvent**(): It is used to create a new events object.
* **createTextNode**(): It is used to create a textnode.
* **execCommand**(): It is used to execute a command specified by the user on the editable selected section. It returns a Boolean value.
* **fullscreenEnabled**(): It is used to check whether the document can be viewed in fullscreen mode or not. It returns a boolean value.
* **getElementById**(): It returns the object of the given ID. If no object with that id exists then it returns null.
* **getElementsByClassName**(): It returns an object containing all the elements with the specified class names in the document as objects.
* **getElementsByName**(): It returns an object containing all the elements with the specified name in the document as objects.
* **getElementsByTagName**(): It returns an object containing all the elements with the specified tag names in the document as objects.
* **hasFocus**(): It returns a boolean value that indicates whether the document or element has focus or not.
* **importNode**(): It imports the copy of a node from another document in the current document.
* **normalize**(): It flushes out the empty nodes and merges the adjacent text nodes with the first text node and
* **normalizeDocument**(): It is used to normalize an HTML document by removing any empty text nodes and joining the adjacent text nodes.
* **open**(): It is used to open the output stream to collect the output.
* **querySelector**(): It returns the first element that matches a specified CSS selector(s) in the document.
* **querySelectorAll**(): It returns a collection of an element’s child elements that matches a specified CSS selector(s) in the document
* **removeEventListener**(): It removes the event handler from an element that has an attached event.
* **renameNode**(): It is used to rename the node.
* **write**(): It is used to write some content or javascript code in the document.
* **writeln**(): It is used to write a document with a newline character after each statement.

**EXAMPLE:**

**­­­­**<!DOCTYPE html>

<html>

<head>

<title>document's Properties</title>

<style>

h1 {

color: green;

}

</style></head>

<body>

<h1> GeeksforGeeks</h1>

<button onclick="myFunction()">CLICK ME</button>

<p id="demo"></p>

<script>

function myFunction() {

var title = document.title;

var domain = document.domain;

var body = document.body;

document.getElementById("demo").innerHTML =

"the title of the document is : "

+ title

+ "<br>"

+ "domain : "

+ domain

+ "<br>"

+ "body : "

+ body;

}

</script>

</body>

</html>

**Window Object**: The window object is the topmost object of the DOM hierarchy. It represents a browser window or frame that displays the contents of the webpage. Whenever a window appears on the screen to display the contents of the document, the window object is created.

**Syntax**: window.property\_name;

**Properties of the window:**

* **Closed**: It holds a Boolean value that represents whether the window is closed or not.
* **console**: It returns a reference to the console object which provides access to the browser’s debugging console.
* **defaultStatus**: It is used to define the default message that will be displayed in the status bar when no activity is carried on by the browser.
* **controllers**: It returns the XUL controller objects for the current Chrome window.
* **customElements**: It returns a reference to the CustomElementRegistry object, which can be used to register new custom elements and also get information about already registered custom elements.
* **crypto**: It returns the browser crypto object.
* **devicePixelRatio**: It returns the ratio between physical pixels and device-independent pixels in the current display.
* **Document**: It returns a reference to the document object of that window.
* **DOMMatrix**: It returns a reference to a DOMMatrix object, which represents 4×4 matrices, suitable for 2D and 3D operations.
* **frames**[]: It represents an array that contains all the frames of a given window.
* **DOMPoint**: It returns a reference to a DOMPoint object, which represents a 2D or 3D point in a coordinate system.
* **History**: It provides information on the URLs visited in the current window.
* Length: It represents the number of frames in the current window.
* **DOMRect**: It returns a reference to a DOMRect object, which represents a rectangle.
* **fullScreen**: This property indicates whether the window is displayed on full screen or not.
* **Location**: It contains the URL of the current window.
* **innerHeight**: It is used to get the height of the content area of the browser window.
* **innerWidth**: It is used to get the width of the content area of the browser window.
* **Name**: It contains the name of the referenced window.
* **Window**: It returns the current window or frame.
* **Navigator**: It returns a reference to the navigator object.
* **outerHeight**: It will get the height of the outside of the browser window.
* **outerWidth**: It will get the width of the outside of the browser window.
* **Status**: It overrides the default status and places a message in the status bar.
* **Top**: It returns a reference to the topmost window containing a frame if many windows are opened.
* **Toolbar**: It will result in the toolbar object, whose visibility can be toggled in the window.
* **Opener**: It contains a reference to the window that opened the current window.
* **Parent**: It refers to the frameset in which the current frame is contained.
* **Screen**: It refers to the screen object
* **Self**: It provides another way to refer to the current window.

**EXAMPLE**

<!DOCTYPE html>

<html>

<head>

<title> Window's Properties</title>

<style>

h1 {

color: green;

}

</style>

</head>

<body>

<h1>GeeksforGeeks</h1>

<button onclick="show()">Check</button>

<p id="prop"></p>

<script>

function show() {

var h = window.innerHeight;

var w = window.innerWidth;

var l = window.location;

var c = window.closed;

document.getElementById("prop").innerHTML =

"Frame's Height: "

+ h + "<br>"

+ "Frame's Width: "

+ w + "<br>"

+ "Window location:"

+ l

+ "<br>"

+ "Window Closed: "

+ c;

}

</script>

</body>

</html>

**Difference between document and window:**

| **document** | **window** |
| --- | --- |
| It represents any HTML document or web page that is loaded in the browser. | It represents a browser window or frame that displays the contents of the webpage. |
| It is loaded inside the window. | It is the very first object that is loaded in the browser. |
| It is the object of window property. | It is the object of the browser. |
| All the tags, elements with attributes in HTML are part of the document. | Global objects, functions, and variables of JavaScript are members of the window object. |
| We can access the document from a window using the window. document | We can access the window from the window only. i.e. window.window |
| The document is part of BOM (browser object model) and dom (Document object model) | The window is part of BOM, not DOM. |
| Properties of document objects such as title, body, cookies, etc can also be accessed by a window like this window. document.title | Properties of the window object cannot be accessed by the document object. |
| syntax:        document.propertyname; | syntax:  window.propertyname; |
| example:       document.title :  will return the title of the document | example:  window.innerHeight : will return the height of the content area of the browser |