**Module (JAVASCRIPT BASIC & DOM) – 4**

1. What Is Javascript?

ANS - JavaScript is a scripting language that enables you to create dynamically updating content, control multimedia, animate images, and pretty much everything else.

2. What Is the Use Of ‘’ isNaN’’ Function?

ANS - The **isNaN** function in JavaScript is used to determine whether a value is NaN (Not-a-Number) or not. It returns a Boolean value indicating whether the provided value is NaN.

3. What Is Negative Infinity?

ANS. NEGATIVE\_INFINITY is a special numeric value that is returned when an arithmetic operation or mathematical function generates a negative value greater than the largest representable number in JavaScript (i.e., more negative than -Number.MAX\_VALUE).

4. Which company developed JavaScript?

ANS. JavaScript was developed by Brendan Eich, who was a Netscape programmer. Brendan Eich developed this new scripting language in just ten days in the year September 1995. At the time of its launch, JavaScript was initially called Mocha. After that, it was called Live Script and later known as JavaScript.

5. What are undeclared and undefined variables?

ANS. Undeclared Variable:

An undeclared variable in JavaScript is a variable that has not been declared using the var, let, or const keyword before it is used. If you try to reference a variable that has not been declared, JavaScript will throw a ReferenceError.

Example of an undeclared variable:

console.log(x); // ReferenceError: x is not defined

Undefined Variable:

An undefined variable in JavaScript is a variable that has been declared, but it has not been assigned a value. When you access the value of an uninitialized variable, JavaScript returns the special value undefined.

Example of an undefined variable:

var y;

console.log(y); // Outputs: undefined

6. Write the code for adding new elements dynamically?

ANS. New elements can be dynamically created in JavaScript with the help of createElement() method. The attributes of the created element can be set using the setAttribute() method.

Syntax:

document.createElement("<tagName>");

// Where <tagName> can be any HTML

// tagName like div, ul, button, etc.

// newDiv element has been created

For Eg: let newDiv = document.createElement("div");

7. What is the difference between ViewState and SessionState?

| **View State** | **SessionState** |
| --- | --- |
| Maintained at page level only. | Maintained at session level. |
| View state can only be visible from a single page and not multiple pages. | Session state value availability is across all pages available in a user session. |
| It will retain values in the event of a postback operation occurring. | In session state, user data remains in the server. Data is available to user until the browser is closed or there is session expiration. |
| Information is stored on the client’s end only. | Information is stored on the server. |
| Used to allow the persistence of page-instance-specific data. | Used for the persistence of user-specific data on the server’s end. |
| ViewState values are lost/cleared when new page is loaded. | SessionState can be cleared by programmer or user or in case of timeouts. |

8. What is === operator?

ANS. JavaScript Strict Equality Operator is used to compare two operands and return true if both the value and type of operands are the same. Since type conversion is not done, so even if the value stored in operands is the same but their type is different the operation will return false.

Syntax a===b

Example :

let a = 2, b=2, c=3;

let d = {name:"Ram"};

let e = {name:"Ram"};

Let f = e;

console.log (a===b); True

console.log(a===c); False

console.log(d===e); False

console.log(f===e); True

Output: Just like other comparisons when two objects are compared their reference is checked and true is only returned if the reference is the same.

9. How can the style/class of an element be changed?

ANS.

Approach 1: Changing CSS with the help of the style property:

Syntax:

document.getElementById("id").style.property = new\_style

Approach 2: Changing the class itself – We can use two properties that can be used to manipulate the classes.

The classList Property: The classList is a read-only property that returns the CSS class names of an element as a DOMTokenList object.

Syntax:

document.getElementById("id").classList

The className Property: This property is used to set the current class of the element to the specified class.

Syntax:

document.getElementById("id").className = class

10. How to read and write a file using JavaScript?

ANS. The read and write operations in a file can be done by using some commands. But the module which is required to perform these operations is to be imported. The required module is **'fs'** which is called as File System module in JavaScript.

**Write operation on a file**

After the File System file is imported then, the **writeFile()** operation is called. The **writeFile()** method is used to write into the file in JavaScript.

**Syntax**

writeFile(path,inputData,callBackFunction)

**Reading from the file**

After the File System module is imported, the reading of the file in JavaScript can be done by using the readFile() function.

**Syntax**

readFile(path, format, callBackFunc)

11. What are all the looping structures in JavaScript?

ANS. Loops can execute a block of code a number of times.

* for - loops through a block of code a number of times
* for/in - loops through the properties of an object
* for/of - loops through the values of an iterable object
* while - loops through a block of code while a specified condition is true
* do/while - also loops through a block of code while a specified condition is true

12. How can you convert the string of any base to an integer in JavaScript?

ANS In JavaScript parseInt() function (or a method) is used to convert the passed-in string parameter or value to an integer value itself. This function returns an integer of the base which is specified in the second argument of the parseInt() function.

Syntax:

parseInt(Value, radix)

13. What is the function of the delete operator?

ANS. The delete operator removes a property from an object. If the property's value is an object and there are no more references to the object, the object held by that property is eventually released automatically.

Syntax

delete object.property

delete object[property]

[Object](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/delete#object) : The name of an object, or an expression evaluating to an object.

[Property](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/delete#property) : The property to delete.

14. What are all the types of Pop up boxes available in JavaScript?

ANS. JavaScript has three kind of popup boxes: Alert box, Confirm box, and Prompt box.

Syntax

* + window.alert("sometext");
  + window.confirm("sometext");
  + window.prompt("sometext","defaultText");

15. What is the use of Void (0)?

ANS. The use of void(0) is often seen in JavaScript, and its primary purpose is to prevent the browser from navigating to a new page when clicking on a link or button. It's commonly used in the href attribute of an <a> (anchor) tag or in the onclick attribute of a button.

16. How can a page be forced to load another page in JavaScript?

ANS. In JavaScript, we can use window.location object to force a page to load another page. We can use the location object to set the URL of a new page.

Syntax

window.location.href = "new\_url";

17.What are the disadvantages of using innerHTML in JavaScript?

ANS.

1. **Security Risk:** Using **innerHTML** to put user-provided content directly into your webpage can open the door to security risks. It might allow attackers to inject harmful code.
2. **Performance:** When dealing with lots of content, changing **innerHTML** might be slower than other methods. It could impact how fast your webpage loads and responds.
3. **Event Handling Issues:** If you change content using **innerHTML**, you might lose existing functionality tied to elements, like click events.

// Existing event handler is lost

document.getElementById("myElement").innerHTML = "<button onclick='myFunction()'>Click me</button>";

1. **Parsing Errors:** Mistakes in your HTML might not be caught and can lead to unexpected behavior.

// Syntax error in HTML string may not be obvious

document.getElementById("myElement").innerHTML = "<p>Missing closing tag<p>";

1. **Accessibility:** Changes made with **innerHTML** might not be as easily understood by assistive technologies, potentially impacting accessibility.