

Module (JAVASCRIPT BASIC & DOM) – 4

(Basic logic Question)

1. What is JavaScript? How to use it?

JavaScript is a high-level, interpreted programming language that is primarily used to add interactivity and dynamic behavior to web pages.

To use JavaScript:

You can include JavaScript code directly within HTML using <script> tags.

External JavaScript files with a .js extension can be linked to HTML using the <script> tag's src attribute..

2. How many types of variables in JavaScript?

There are three types of variables in JavaScript: var, let, and const. These keywords are used to declare variables.

var: Function-scoped or globally scoped variable. It is outdated JavaScript.

let: Block-scoped variable, introduced in ECMAScript 6 (ES6). It has a limited scope within the block, statement, or expression where it is used.

const: Also introduced in ES6, it declares a constant variable it is not change.

3. Define Data Types in JavaScript?

JavaScript has several data types, which can be categorized as primitive and non-primitive.

Primitive data types:

String: Textual data.

Number: Numeric data, including integers and floating-point numbers.

Boolean: Represents either true or false.

Undefined: Default value of variables that have not been assigned a value.

Null: Represents the intentional absence of any object value.

Non-primitive data type:

Object: A collection of key-value pairs, often used to represent complex entities.

Array : Is a collection of Data type. []

Date : new Date()...

4. Write a mul function which will work properly when invoked with the following syntax:

`console.log(mul(2)(3)); // Output: 6...`

```
function mul(x) {  
    return function(y) {  
        return x * y;  
    };  
}
```

5. What is the difference between undefined and undeclared in JavaScript?

Undefined: A variable is declared but has not been assigned a value. It is a special value that represents the absence of a value or an uninitialized variable.

Undeclared: A variable that has not been declared using var, let, or const. Accessing an undeclared variable usually results...

6. Using `console.log()` print out the following statement:

"The quote 'There is no exercise better for the heart than reaching down and lifting people up.' by John Holmes teaches us to help one another."

Using `console.log()` to print the quote by Mother Teresa:

Ex:-

```
console.log("The quote 'There is no exercise better for the heart");  
console.log("Spread love everywhere you go. other Teresa");
```

7. Check if `typeof '10'` is exactly equal to 10. If not, make it exactly equal?

This is Check `typeof 10` code....and exactly to 10.perfect code..

```

if (typeof '10' === 'string') {
    console.log("They are exactly equal.");
} else {
    console.log("They are not exactly equal.");
    // Convert '10' to a number
    var convertedNumber = parseInt('10');
    console.log(convertedNumber); // Output: 10
}

```

8. Write a JavaScript program to find the area of a triangle?
 Maths Formula $(a+b+c)/2$; $\text{Math.sqrt}(p*((p-a)*(p-b)*(p-c)))$; Code..

```

function calculateTriangleArea(base, height) {
    var area = 0.5 * base * height;
    return area;
}

var triangleArea = calculateTriangleArea(5, 8);
console.log("The area of the triangle is: " + triangleArea);

```

9.
 10. Write a JavaScript program to calculate days left until next Christmas?

```
var today = new Date();
var nextChristmas = new Date(today.getFullYear(), 11, 25); // Christmas is
// If Christmas has already occurred this year, set it for next year
if (today > nextChristmas) {
    nextChristmas.setFullYear(today.getFullYear() + 1);
}

// Calculate the difference in days
var daysLeft = Math.ceil((nextChristmas - today) / (1000 * 60 * 60 * 24));

console.log("Days left until next Christmas: " + daysLeft);
```

11. What is a Condition Statement?

A condition statement in programming is a construct that allows you to execute different blocks of code based on whether a specified condition evaluates to true or false. In JavaScript, common condition statements include if, else if, and else.

```
var x = 10;

if (x > 0) {
    console.log("x is positive");
} else if (x < 0) {
    console.log("x is negative");
} else {
    console.log("x is zero");
}
```

12. Find the circumference of a Rectangle with the formula $C = 2 * (\text{length} + \text{width})$?

```
function circum(length, width) {  
    var circumference = 2 * (length + width);  
    return circumference;  
}  
  
var rectangle = circum(4, 6);  
console.log("The circumference of the rectangle is: " + rectangle);
```

13. Write a program to convert years into days and days into years:

This is code for convert year to days...

```
function fullyear(input) {  
    var daysInYear = 365;  
  
    if (typeof input === 'number') {  
        var years = Math.floor(input);  
        var days = Math.floor((input % 1) * daysInYear);  
  
        console.log(years + " years is approximately " + (years * daysInYear + days) + " days.");  
    } else {  
        console.log("Invalid input. Please provide a numeric value.");  
    }  
}  
  
fullyear(3.5); // Example usage
```

14. Convert temperature Fahrenheit to Celsius with conditional logic:

```
function convert(fahrenheit) {
    var celsius = (fahrenheit - 32) * 5 / 9;
    return celsius;
}

var a = 98;
var b = convert(a);
console.log("Temperature in Celsius: " + b.toFixed(2) + "°C");
```

15. Get the extension of a filename:

This is use function filename and return filename...

```
function filename(filename) {
    return filename.split('.').pop();
}

var a = "example.txt";
var b = filename(a);
console.log("File extension: " + b);
```

16. Result of the expression (5 > 3 && 2 < 4):

The result is true because both conditions are true.

17. Result of the expression (true && 1 && "hello"):

The result is "hello" because all conditions are truthy, and the last truthy value is returned.

18. Result of the expression true && false || false && true:

The result is false because the expression is evaluated as (true && false) || (false && true).

19. What is a Loop and Switch Case in JavaScript?

Loop: A loop is a programming construct that allows you to repeat a set of instructions until a certain condition is met.

Switch Case: switch is a control flow statement used in JavaScript to perform different actions based on different conditions. It is a cleaner way to write multiple if-else statements.

20. What is the use of is NaN function?

The isNaN function in JavaScript is used to determine whether a value is NaN (Not-a-Number). It returns true if the value is NaN, and false otherwise. It's often used to check..

21. Difference between && and || in JavaScript:

&& (Logical AND): Returns true if both operands are true.

|| (Logical OR): Returns true if at least one of the operands is true.

22. What is the use of void(0)?

void(0) is often used in JavaScript to create an expression that evaluates to undefined. It is commonly used as the href attribute in anchor tags (<a> elements)..

23. Check if a Number Is Positive or Negative in JavaScript.

```
function checkPositiveNegative(number) {  
  if (number > 0) {  
    console.log("Positive");  
  } else if (number < 0) {  
    console.log("Negative");  
  } else {  
    console.log("Zero");  
  }  
}  
  
checkPositiveNegative(5); // Example usage
```


24. Check if a Character Is a Vowel or Not:

```
function isVowel(char) {  
    var vowels = "aeiouAEIOU";  
    if (vowels.includes(char)) {  
        console.log(char + " is a vowel");  
    } else {  
        console.log(char + " is not a vowel");  
    }  
}  
  
isVowel('a'); // Example usage
```

25. Check whether a number is negative, positive, or zero:

```
function checkNumberType(number) {  
    if (number > 0) {  
        console.log("Positive");  
    } else if (number < 0) {  
        console.log("Negative");  
    } else {  
        console.log("Zero");  
    }  
}  
  
checkNumberType(-3); // Example usage
```


26. Check if a number is even or odd using ternary operator in JS:

```
function checkEvenOdd(number) {  
    var result = number % 2 === 0 ? "Even" : "Odd";  
    console.log(result);  
}  
  
checkEvenOdd(7); // Example usage
```

27. Find the maximum number among 3 numbers using ternary operator in JS:

```
function findMaxNumber(a, b, c) {  
    var max = (a > b) ? (a > c ? a : c) : (b > c ? b : c);  
    console.log("Maximum number is: " + max);  
}  
  
findMaxNumber(5, 8, 3); // Example usage
```

28. Find the minimum number among 3 numbers using ternary operator in JS:

```
function findMinNumber(a, b, c) {  
    var min = (a < b) ? (a < c ? a : c) : (b < c ? b : c);  
    console.log("Minimum number is: " + min);  
}  
  
findMinNumber(5, 8, 3); // Example usage
```

29. Find the largest of three numbers in JS:

```
function findLargestNumber(a, b, c) {  
    var max = Math.max(a, b, c);  
    console.log("Largest number is: " + max);  
}  
  
findLargestNumber(5, 8, 3); // Example usage
```

29. Switch case to show:

i. Monday to Sunday using switch case in JS.

```
function getDayName(day) {  
    switch (day) {  
        case 1:  
            console.log("Monday");  
            break;  
        case 2:  
            console.log("Tuesday");  
            break;  
        case 3:  
            console.log("Wednesday");  
            break;  
        case 4:  
            console.log("Thursday");  
            break;  
        case 5:  
            console.log("Friday");  
            break;  
        case 6:  
            console.log("Saturday");  
            break;  
        case 7:  
            console.log("Sunday");  
            break;  
        default:  
            console.log("Invalid day");  
    }  
}
```

30. What are the looping structures in JavaScript? Any one Example?
for loop:

```
for (let i = 0; i < 5; i++) {  
  console.log(i);  
}
```

while loop:

```
let i = 0;  
while (i < 5) {  
  console.log(i);  
  i++;  
}
```

31. Write a print 972 to 897 using for loop in JS:

```
for (let i = 972; i >= 897; i--) {  
  console.log(i);  
}
```

32. Write to print factorial of given number:

```
function factorial(n) {  
  let result = 1;  
  for (let i = 1; i <= n; i++) {  
    result *= i;  
  }  
  return result;  
}  
  
console.log(factorial(5)); // Exa
```

33. Write to print Fibonacci series up to given numbers:

```
function fibonacciSeries(n) {  
  let fib = [0, 1];  
  
  for (let i = 2; fib[i - 1] + fib[i - 2] <= n; i++) {  
    fib[i] = fib[i - 1] + fib[i - 2];  
  }  
  
  console.log(fib.join(', '));  
}  
  
fibonacciSeries(50); // Example for Fibonacci series u
```

34. Write to print number in reverse order e.g.: number = 64728 ---> reverse = 82746 in JS:

```
function reverseNumber(num) {  
  let reversed = 0;  
  while (num > 0) {  
    reversed = reversed * 10 + (num % 10);  
    num = Math.floor(num / 10);  
  }  
  console.log(reversed);  
}  
  
reverseNumber(64728); // Example for revers
```

34.

35. Write a program to make a summation of given number (E.g., 1523 Ans: - 11) in JS:

```
function sumOfDigits(number) {  
    let sum = 0;  
  
    // Iterate through each digit of the number  
    while (number !== 0) {  
        sum += number % 10;  
        number = Math.floor(number / 10);  
    }  
  
    console.log("Sum of digits:", sum);  
}  
  
sumOfDigits(1523); // Example for the number 1523
```

36. Write a program to make a summation of the first and last digit. (E.g., 1234 Ans: - 5) in JS:-

```
function sum(number) {  
    const lastDigit = number % 10;  
  
    // Find the first digit  
    while (number >= 10) {  
        number = Math.floor(number / 10);  
    }  
    const firstDigit = number;  
  
    const sum = firstDigit + lastDigit;  
    console.log("Sum of first and last digit:", sum);  
}  
  
sum(1234); // Example for the number 1234
```

37. Use console.log() and escape characters to print the following pattern in JS:-

```
for (let i = 1; i <= 5; i++) {  
  let output = `${i} `;  
  let exponent = 1;  
  
  for (let j = 1; j <= 4; j++) {  
    exponent *= i;  
    output += `${exponent} `;  
  }  
  
  console.log(output);  
}
```

38. Use pattern in console.log in JS?

1.Pattern:-

```
for (let i = 1; i <= 5; i++) {  
  let row = '';  
  for (let j = 1; j <= i; j++) {  
    row += (j % 2 === 0) ? '0 ' : '1 ';  
  }  
  console.log(row.trim());  
}
```


2.Pattern:-

```
let charCode = 65;
for (let i = 1; i <= 5; i++) {
  let row = '';
  for (let j = 1; j <= i; j++) {
    row += String.fromCharCode(charCode) + ' ';
    charCode++;
  }
  console.log(row.trim());
}
```

3.Pattern:-

```
let num = 1;
for (let i = 1; i <= 5; i++) {
  let row = '';
  for (let j = 1; j <= i; j++) {
    row += num + ' ';
    num++;
  }
  console.log(row.trim());
}
```

4.pattern:-

```
for (let i = 1; i <= 5; i++) {
  let row = '';
  for (let j = 1; j <= i; j++) {
    row += '* ';
  }
  console.log(row.trim());
}
```


39. Accept 3 numbers from user using while loop and check each numbers palindrome?

```
const readline = require('readline');

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout
});

let count = 0;

function isPalindrome(num) {
  const originalNum = num;
  let reversedNum = 0;

  while (num > 0) {
    const digit = num % 10;
    reversedNum = reversedNum * 10 + digit;
    num = Math.floor(num / 10);
  }

  return originalNum === reversedNum;
}
```

(Array and object Question)

40. Write a JavaScript Program to display the current day and time in the following format.

Sample Output: Today is Friday. Current Time is 12 PM: 12 : 22 2 ?

```
const today = new Date();
const daysOfWeek = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"];
const day = daysOfWeek[today.getDay()];

let hours = today.getHours();
const ampm = hours >= 12 ? 'PM' : 'AM';
hours = hours % 12 || 12;

const minutes = today.getMinutes();
const seconds = today.getSeconds();

console.log(`Today is ${day}. Current Time is ${hours} ${ampm}: ${minutes} : ${seconds}`);
```

41. Write a JavaScript program to get the current date?

```
const currentDate = new Date();  
console.log(`Current Date: ${currentDate.toString()}`);
```

42. Write a JavaScript program to compare two objects?

```
const obj1 = { a: 1, b: 2, c: 3 };  
const obj2 = { a: 1, b: 2, c: 3 };  
  
const areObjectsEqual = JSON.stringify(obj1) === JSON.stringify(obj2);  
console.log(`Are the objects equal? ${areObjectsEqual}`);
```

43. Write a JavaScript program to convert an array of objects into CSV string?

```
const arrayOfObjects = [  
  { name: 'John', age: 30, city: 'New York' },  
  { name: 'Alice', age: 25, city: 'San Francisco' },  
  { name: 'Bob', age: 35, city: 'Los Angeles' }  
];  
  
const csvString = arrayOfObjects.map(obj => Object.values(obj).join(',')).join('\n');  
console.log(csvString);
```

44. Write a JavaScript program to capitalize first letter of a string?

```
const capitalizeFirstLetter = (str) => str.charAt(0).toUpperCase() + str.slice(1);

const inputString = 'hello world';
const capitalizedString = capitalizeFirstLetter(inputString);
console.log(capitalizedString);
```

45. Write a JavaScript program to determine if a variable is array?

```
const isArray = (variable) => Array.isArray(variable);

const exampleArray = [1, 2, 3];
const exampleVariable = 'Not an array';

console.log(`Is exampleArray an array? ${isArray(exampleArray)}`);
console.log(`Is exampleVariable an array? ${isArray(exampleVariable)}`);
```

46. Write a JavaScript program to clone an array?

```
const originalArray = [1, 2, 3];
const clonedArray = [...originalArray];

console.log('Original Array:', originalArray);
console.log('Cloned Array:', clonedArray);
```

47. What is the drawback of declaring methods directly in JavaScript objects?

The main drawback is that methods declared directly in JavaScript objects are duplicated for each instance of the object, which consumes more memory. If the method is the same for all instances, it's more memory-efficient to declare it in the object's prototype.

48. Print the length of the string on the browser console:

```
const myString = 'Hello, World!';  
console.log(myString.length);
```

49. Change all string characters to capital letters:

```
const lowercaseString = 'hello, world!';  
const uppercaseString = lowercaseString.toUpperCase();  
console.log(uppercaseString);
```

50. Drawback of declaring methods directly in JavaScript objects:

As mentioned earlier, the main drawback is that methods declared directly in JavaScript objects are duplicated for each instance of the object, which can lead to increased memory consumption. This is not memory-efficient, especially if the methods are the same for all instances. Declaring methods in the prototype allows them to be shared among all instances of the object, reducing memory usage.

51. JavaScript program to get the current date in different formats:

```
const currentDate = new Date();

// Format 1: mm-dd-yyyy
const mmdyyyy = `${currentDate.getMonth() + 1}-${currentDate.getDate()}-${currentDate.getFullYear()}`;

// Format 2: mm/dd/yyyy
const mmdyyyy_slash = `${currentDate.getMonth() + 1}/${currentDate.getDate()}/${currentDate.getFullYear()}`;

// Format 3: dd-mm-yyyy
const ddmmyyy = `${currentDate.getDate()}-${currentDate.getMonth() + 1}-${currentDate.getFullYear()}`;

// Format 4: dd/mm/yyyy
const ddmmyyy_slash = `${currentDate.getDate()}/${currentDate.getMonth() + 1}/${currentDate.getFullYear()}`;

console.log(`mm-dd-yyyy: ${mmdyyyy}`);
console.log(`mm/dd/yyyy: ${mmdyyyy_slash}`);
console.log(`dd-mm-yyyy: ${ddmmyyy}`);
console.log(`dd/mm/yyyy: ${ddmmyyy_slash}`);
```

52. Use indexOf to determine the position of the first occurrence of 'a' in '30 Days Of JavaScript':

```
const str = '30 Days Of JavaScript';
const position = str.indexOf('a');
console.log(`The position of 'a' is: ${position}`);
```

53. Use lastIndexOf to determine the position of the last occurrence of 'a' in '30 Days Of JavaScript':

```
const str = '30 Days Of JavaScript';
const position = str.lastIndexOf('a');
console.log(`The last position of 'a' is: ${position}`);
```


54. Form Validation in JavaScript:

```
// Assuming you have a form with an input field with the id 'username'
const form = document.getElementById('myForm');

form.addEventListener('submit', function (event) {
    const username = document.getElementById('username').value;

    if (username === '') {
        alert('Username cannot be empty');
        event.preventDefault(); // Prevent form submission
    }
});
```

55. Form Validation for Email, Number, and Password in JavaScript:

```
if(pass=="" || pass==null) // for null condition
{
    alert('Please fill out the pass'); // alert msg
    return false; //return false means msg show and again
}

if(!(pass.length >=3 && pass.length <= 8))
{
    alert('Please,provide min 3 & max 8 char in pass');
    return false;
}

var mail=/^[a-zA-Z0-9_]+@[a-zA-Z]+\.[a-zA-Z]{2,4}$/;
if(!mail.test(email))
{
    alert('Please fill proper email id'); // alert msg
    return false; //return false means msg show and again
}

var pno=document.forms["myform"]["pno"].value;
if(pno=="" || pno==null) // for null condition
{
    alert('Please fill out the pno'); // alert msg
    return false; //return false means msg show and again
}
```

56. Dynamic Form Validation in JS?

```
<script>
const form = document.getElementById("myForm");

form.addEventListener("input", function(event) {
  const usernameInput = document.getElementById("username");
  const passwordInput = document.getElementById("password");
  const usernameError = document.getElementById("usernameError");
  const passwordError = document.getElementById("passwordError");

  if (usernameInput.value.length < 5) {
    usernameError.textContent = "Username must be at least 5 characters";
  } else {
    usernameError.textContent = "";
  }

  if (passwordInput.value.length < 8) {
    passwordError.textContent = "Password must be at least 8 characters";
  } else {
    passwordError.textContent = "";
  }
});
</script>
```

57. How many types of JS events? How to use them?

Mouse events (click, mouseover, mouseout, etc.)

Keyboard events (keydown, keyup, keypress)

Form events (submit, change, input)

Document events (DOMContentLoaded, load, unload)

```
document.getElementById("myButton").addEventListener("click", function() {
  // Your code here
  alert("Button clicked!");
});
```

58. What is BOM vs DOM in JS?

DOM: It represents the structure of a document as a tree of objects. The DOM is used to interact with HTML or XML documents.

BOM: It represents the browser as an object. It includes objects like window, navigator, location, screen, and document (which is part of the DOM).

59. Array vs Object differences in JS?

Arrays are ordered collections of values, accessed by numerical indices

```
document.getElementById("myButton").addEventListener("click", function() {  
  // Your code here  
  alert("Button clicked!");  
});
```

Objects are unordered collections of key-value pairs.

```
const fruits = ["apple", "banana", "orange"];
```

60. Split the string into an array using split() Method?

The split() method in JavaScript is used to split a string into an array of substrings based on a specified delimiter.

```
const sentence = "Hello, world!";  
const words = sentence.split(" ");  
  
console.log(words); // Output: ["Hello,", "world!"]
```

61. Check if the string contains a word "Script" using includes() method?

The includes() method checks if a string contains a specific substring and returns a boolean value.

```
const sentence = "30 Days of JavaScript";

if (sentence.includes("Script")) {
  console.log("The string contains the word 'Script'");
} else {
  console.log("The string does not contain the word 'Script'");
}
```

62. Change all the string characters to lowercase letters using toLowerCase() method. The toLowerCase() method in JavaScript is used to convert all characters in a string to lowercase

```
const mixedCaseString = "Hello, World!";
const lowercaseString = mixedCaseString.toLowerCase();

console.log(lowercaseString); // Output: "hello, world!"
```

63. What is the character at index 15 in the '30 Days of JavaScript' string? Use charAt() method. The charAt() method in JavaScript is used to get the character at a specified index in a string

```
const sentence = "30 Days of JavaScript";
const characterAtIndex15 = sentence.charAt(15);

console.log(characterAtIndex15); // Output: "v"
```

64. Copy one string to another string in JS?

```
const originalString = "Hello, World!";  
let copiedString = originalString;  
  
console.log(copiedString); // Output: "Hello, World!"
```

65. Find the length of a string without using the library function?

```
function findStringLength(inputString) {  
  let length = 0;  
  
  for (let char of inputString) {  
    length++;  
  }  
  
  return length;  
}  
  
const myString = "Hello, World!";  
const lengthWithoutLibraryFunction = findStringLength(myString);  
  
console.log(lengthWithoutLibraryFunction); // Output: 13
```

- What is JavaScript?
JavaScript is a high-level, interpreted programming language that is widely used for creating dynamic content on the web. It is primarily known for adding interactivity to web pages.
- What is the use of isNaN function?
The isNaN function in JavaScript is used to check whether a value is NaN (Not-a-Number) or not. It returns true.
isNaN(123); // false
isNaN('Hello'); // true
- What is negative Infinity?
Negative Infinity is a special value in JavaScript representing negative infinity. It is the result of dividing a negative number by zero or any mathematical operation that results in an infinitely negative value.

- Which company developed JavaScript?

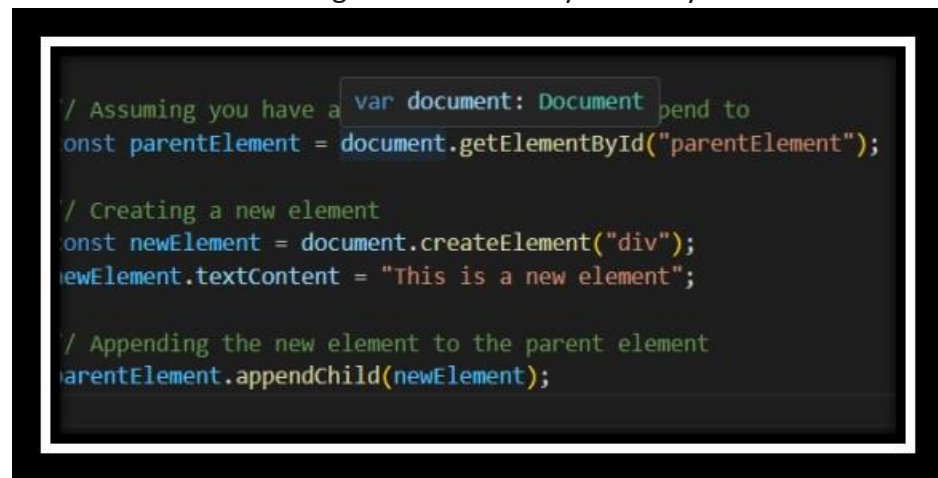
JavaScript was developed by Netscape Communications Corporation, with Brendan Eich creating the language in 1995.

- What are undeclared and undefined variables?

Undeclared variables: Variables that have been neither declared using the var, let, or const keyword nor defined in the current scope. Accessing an undeclared variable will result in a ReferenceError.

Undefined variables: Variables that have been declared but not assigned a value are considered undefined.

- Write the code for adding new elements dynamically?



```

/ Assuming you have a var document: Document
const parentElement = document.getElementById("parentElement");

/ Creating a new element
const newElement = document.createElement("div");
newElement.textContent = "This is a new element";

/ Appending the new element to the parent element
parentElement.appendChild(newElement);

```

- What is the difference between ViewState and SessionState?

ViewState: ViewState is used to store the state of the page and its controls between postbacks. It is stored as a hidden field on the page.

SessionState: SessionState is used to store user-specific information that persists across multiple requests.

- What is === operator?

The === operator in JavaScript is the strict equality operator. It checks both value and type equality, unlike the loose equality operator (==),

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

```

// Change style
const element = document.getElementById("myElement");
element.style.color = "red";

// Change class
element.className = "newClass";
// OR
element.classList.add("newClass");

```

- How to read and write a file using JavaScript?
In a web browser environment, JavaScript has limited access to the file system due to security reasons. However, you can read files using the FileReader API or write data to a file using server-side technologies like Node.js.

- What are all the looping structures in JavaScript?
JavaScript supports several looping structures:

for loop

while loop

do-while loop

for...in loop (used for iterating over object properties)

for...of loop (used for iterating over iterable objects like arrays)

- How can you convert the string of any base to an integer in JavaScript?
You can use the parseInt function to convert..

```

const binaryString = "1010";
const decimalNumber = parseInt(binaryString, 2);
console.log(decimalNumber); // Output: 10

```

- What is the function of the delete operator?
The delete operator in JavaScript is used to delete an object property or an element at a specified index in an array.
It does not delete variables or functions.

- What are all the types of Pop up boxes available in JavaScript?
JavaScript provides three types of pop-up boxes:

alert() - displays an alert dialog box.

confirm() - displays a dialog box with OK and Cancel buttons.

prompt() - displays a dialog box with an input field

- What is the use of Void (0)?
void(0) is often used in JavaScript to generate an undefined value. It's commonly used in href attributes to prevent the page from navigating..
- How can a page be forced to load another page in JavaScript?
You can use the window.location object to navigate to another page...



- What are the disadvantages of using innerHTML in JavaScript?
Using innerHTML to manipulate or append content can have security implications, as it may execute scripts included in the HTML string. It's also less efficient than alternative methods like createElement and appendChild because it requires the browser to parse and re-render the entire HTML content of an element.

