1 What is JavaScript?

JavaScript is a scripting language used to create and control dynamic website content, i.e. anything that moves, refreshes, or otherwise changes on your screen without requiring you to manually reload a web page. Features like: animated graphics. photo slideshows

2 What is the use of isNaN function?

NaN stands for "Not a Number" and is a value in JavaScript used to represent an undefined or unrepresentable value. NaN is the result of an operation that was supposed to return a number, but couldn't because of an error or undefined/empty value.

3 What is negative Infinity?

NEGATIVE\_INFINITY is a property of the JavaScript Number object.

You can only use it as Number.NEGATIVE\_INFINITY.

Using x.NEGATIVE\_INFINITY, where x is a variable, will return undefined

EXAMPLE :

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <p id="demo"></p>

<script>

let x = 100;

document.getElementById("demo").innerHTML = x.NEGATIVE\_INFINITY;

</script>

</body>

</html>

4 Which company developed JavaScript?

JavaScript was invented by Brendan Eich in 1995 while he was working at Netscape.

5 What are undeclared and undefined variables?

Undefined: It occurs when a variable has been declared but has not been assigned any value. Undefined is not a keyword.

Undeclared: It occurs when we try to access any variable that is not initialized or declared earlier using the var or const keyword. If we use ‘typeof’ operator to get the value of an undeclared variable, we will face the runtime error with the return value as “undefined”. The scope of the undeclared variables is always global.

6 Write the code for adding new elements dynamically?

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content=

        "width=device-width, initial-scale=1.0">

    <style>

        html,

        body {

            height: 100%;

            width: 100%;

        }

        .button {

            display: flex;

            align-items: center;

            justify-content: center;

        }

        .tasks {

            display: flex;

            justify-content: center;

            align-items: center;

            flex-direction: column;

            margin-top: 20px;

        }

    </style>

</head>

<body>

    <div class="button">

        <button id="addTask">Add task</button>

    </div>

    <div class="tasks"></div>

    <script type="text/javascript">

        let task = document.getElementsByClassName("tasks");

        let addTask = document.getElementById("addTask");

        addTask.addEventListener('click', function () {

            for (let i = 0; i < task.length; i++) {

                let newDiv = document.createElement("div");

                newDiv.setAttribute("class", "list");

                newDiv.innerText = "New Div created";

                task[i].append(newDiv);

            }

        })

    </script>

</body>

</html>

7 What is the difference between ViewState and SessionState?

| **ViewState** | **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?

Strict Equality ===

The === operator, also known as the strict equality operator, checks if two values are equal and of the same data type.

**EXAMPLE**

const num1 = 3;

const num2 = '3';

console.log(num1 === num2); // false (strict equality)

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

Changing CSS with the help of the style property:

Syntax:

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

10 How to read and write a file using JavaScript?

Here's an example of how we can read and write a file using JavaScript with

the File System API:

To read a file:

```javascript

function readFile(file) {

const reader = new FileReader();

reader.onload = function(event) {

const contents = event.target.result;

console.log(contents);

};

reader.onerror = function(event) {

console.error("Error reading file:", event.target.error);

};

reader.readAsText(file);

}

// Usage example

const fileInput = document.getElementById('fileInput');

fileInput.addEventListener('change', function(event) {

const file = event.target.files[0];

readFile(file);

});

```

To write to a file:

```javascript

function writeFile(filename, content) {

const element = document.createElement('a');

element.setAttribute('href', 'data:text/plain;charset=utf-8,' +

encodeURIComponent(content));

element.setAttribute('download', filename);

element.style.display = 'none';

document.body.appendChild(element);

element.click();

document.body.removeChild(element);

}

// Usage example

const filename = 'file.txt';

const content = 'Hello, world!';

writeFile(filename, content);

```

In these examples, we're using the FileReader API to read the contents of a file

and the data URL scheme to create a downloadable file for writing. Remember to

handle any errors that may occur during the process.

11 What are all the looping structures in JavaScript?

JavaScript loops are essential for efficiently handling repetitive tasks. They execute a block of code repeatedly as long as a specified condition remains true. These loops are powerful tools for automating tasks and streamlining your code.

For example, suppose we want to print “Hello World” 5 times. This can be done using JS Loop easily. In Loop, the statement needs to be written only once and the loop will be executed 5 times

for Loop

The JS for loop provides a concise way of writing the loop structure. The for loop contains initialization, condition, and increment/decrement in one line thereby providing a shorter, easy-to-debug structure of looping.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <script>

            let x;

            for (x = 2; x <= 4; x++) {

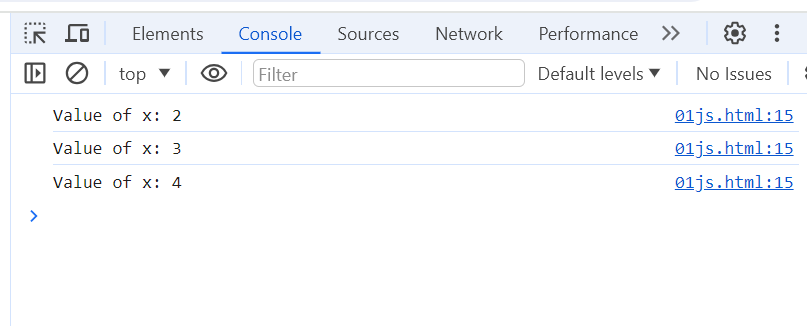
    console.log("Value of x: " + x);

}

     </script>

</body>

</html>



while Loop

The JS while loop is a control flow statement that allows code to be executed repeatedly based on a given Boolean condition. The while loop can be thought of as a repeating if statement.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <script>

let val = 1;

while (val < 6) {

    console.log(val);

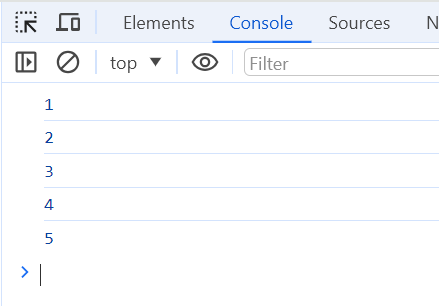
    val += 1;

}

     </script>

</body>

</html>



do-while Loop

The JS do-while loop is similar to the while loop with the only difference is that it checks for the condition after executing the statements, and therefore is an example of an Exit Control Loop. It executes loop content at least once event the condition is false.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <script>

         let test = 1;

do {

    console.log(test);

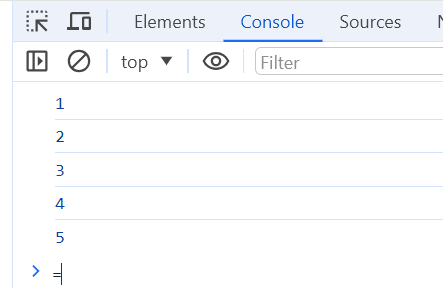
    test++;

} while(test <= 5)

     </script>

</body>

</html>



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

Convert String to an Integer using parseInt() Method

The parseInt() method accepts the string and radix parameter and converts it into an integer.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <script>

        let a = "100";

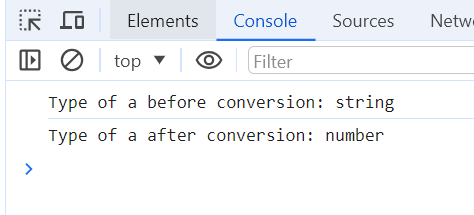
console.log("Type of a before conversion: " + typeof a);

console.log("Type of a after conversion: " + typeof parseInt(a));

     </script>

</body>

</html>



13 • What is the function of the delete operator.

The delete operator in JavaScript is used to remove a property from an object. It works for both properties owned by the object and those inherited from prototypes. When used on an array item, it creates a ‘hole’ in the array.

The delete operator in JavaScript is used to remove a property from an object. It works for both properties owned by the object and those inherited from prototypes. When used on an array item, it creates a ‘hole’ in the array.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <script>

        let emp = {

    firstName: "Raj",

    lastName: "Kumar",

    salary: 40000

}

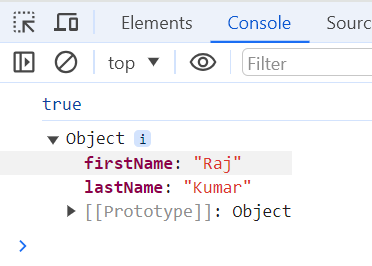
console.log(delete emp.salary);

console.log(emp);

     </script>

</body>

</html>



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

In Javascript, popup boxes are used to display the message or notification to the user.

There are three types of pop-up boxes in JavaScript:

Alert Box

It is used when a warning message is needed to be produced. When the alert box is displayed to the user, the user needs to press ok and proceed.

<!DOCTYPE html>

<html>

<head>

    <title>Pop-up Box type | Alert Box</title>

</head>

<body>

    <center>

        <h1>EMPIRE ANKIT</h1>

        <h3>Alert Box</h3>

        <button onclick="geekAlert()">

            Click here for alert box

        </button>

        <script>

            function geekAlert() {

                alert("ANKIT EMPIRE "

                            +  "BGMI");

            }

        </script>

    </center>

</body>

</html>



Prompt Box

It is a type of pop up box which is used to get the user input for further use. After entering the required details user have to click ok to proceed next stage else by pressing the cancel button user returns the null value.

<!DOCTYPE html>

<html>

<head>

    <title>

        Pop-up Box type | Prompt Box

    </title>

</head>

<body>

    <center>

        <h1>EMPIRE ANKIT</h1>

        <h3>BGMI</h3>

        <input type="button" onclick="geekPrompt();"

        value="Click here for Prompt box"/>

        <script>

            function geekPrompt() {

                let x = prompt("Enter your mail here : ");

                document.write("Your ID : " + x);

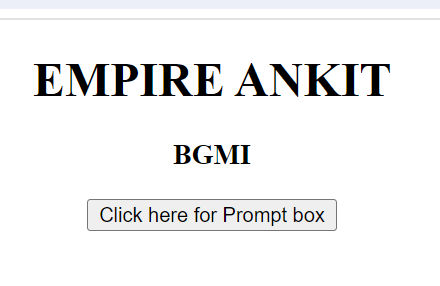
            }

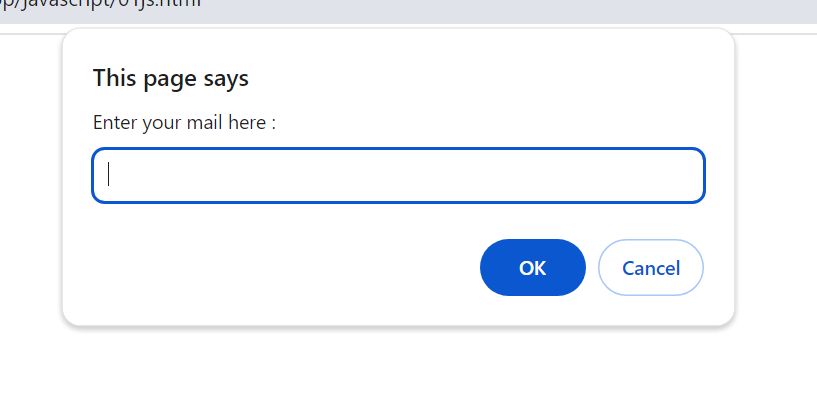
        </script>

    </center>

</body>

</html>

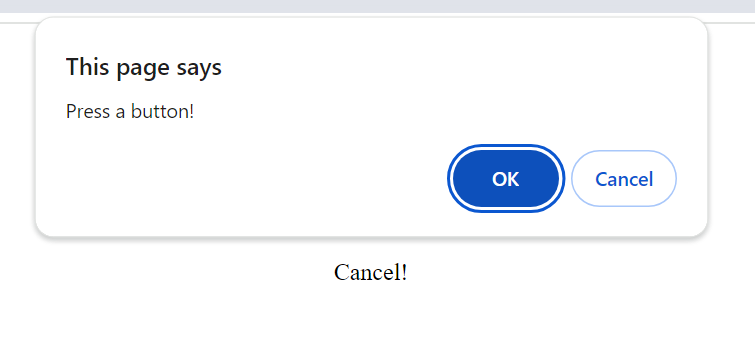




Confirm Box

It is a type of pop-up box that is used to get authorization or permission from the user. The user has to press the ok or cancel button to proceed.





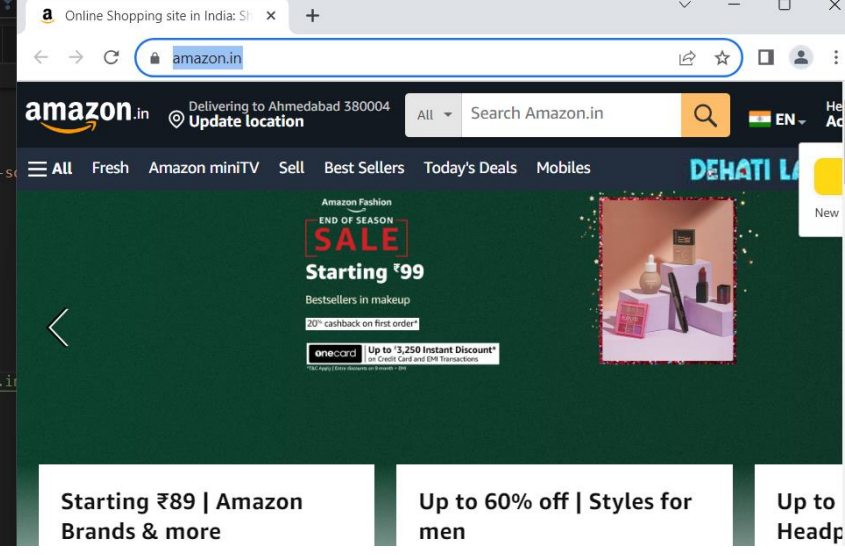
15 What is the use of Void (0)?

The `void(0)` expression in JavaScript is often used in conjunction with the `href` attribute of an anchor tag (``). It is typically used to prevent the default behavior of the anchor tag, which is to navigate to a new page when clicked. By setting the `href` attribute to `javascript:void(0)`, clicking on the anchor tag will not cause any navigation. It essentially does nothing. Here's an example: ```html [Click me](javascript:void(0)) ``` When the user clicks on the "Click me" link, nothing will happen. It's commonly used when you want to attach a click event handler to the anchor tag using JavaScript, but you don't want the page to navigate to a new URL. However, it's worth noting that in modern web development, it's generally recommended to use event listeners and prevent default behavior using JavaScript instead of relying on `void(0)

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

If we want to force a page to load another page in JavaScript.we can use the `window.location` object to change the URL of the current page. Here's an example:





It's important to note that this will immediately navigate the user to the new page, so make sure to use it responsibly and provide a clear indication to the user that they will be redirected.

17 What are the disadvantages of using innerHTML in JavaScript?

Using `innerHTML` in JavaScript does have some disadvantages to consider: 1. \*\*Security risks\*\*: When using `innerHTML`, we need to be cautious about potential security vulnerabilities like cross-site scripting (XSS) attacks. If we insert untrusted or user-generated content directly into `innerHTML`, it can execute malicious scripts. 2. \*\*Performance impact\*\*: Updating `innerHTML` can be less efficient compared to other DOM manipulation methods. When we modify `innerHTML`, the browser needs to re-parse and re-render the entire HTML structure of the affected element, which can be slower for large or complex content. 3. \*\*Event listener loss\*\*: If we update an element's `innerHTML`, any event listeners attached to its child elements will be lost. we'll need to reattach the event listeners after each `innerHTML` update. To illustrate, here's an example: ```javascript // Example using innerHTML const container = document.getElementById("container"); container.innerHTML = "

Hello, **world!**

"; // Potential issues: // - Security risks if untrusted content is inserted. // - Performance impact due to re-parsing and re-rendering. // - Event listeners attached to child elements are lost and need to be reattached. ``` To mitigate these disadvantages,we can explore alternative approaches like using DOM manipulation methods (`createElement`, `appendChild`, etc.) or utilizing modern frameworks like React or Vue.js that handle updates efficiently and provide security measures