•What is JavaScript?

JavaScript is a high-level programming language that is commonly used in web development to add interactive and dynamic features to websites. It is a versatile language that can be used both on the client-side (in the user's browser) and on the server-side (using platforms such as Node.js).

• What is the use of isNaN function?

The `isNaN()` function in JavaScript is used to determine whether a value is not a number (NaN) or not. This function takes one argument and returns a boolean value (either `true` or `false`) indicating whether the argument is NaN or not.

The `isNaN()` function is often used in situations where a value needs to be checked to ensure that it is a valid number. For example, when processing user input in a web form, the `isNaN()` function can be used to validate numeric input before it is submitted to the server.

• What is negative Infinity?

In JavaScript, `Infinity` is a numeric value that represents positive infinity, which is a value that is greater than any other number. However, JavaScript also has a special value called negative infinity, which is represented by the constant `-Infinity`.

Negative infinity represents a value that is less than any other number, including negative numbers. It is often used to represent the result of a mathematical operation that cannot be represented as a finite number.

• Which company developed JavaScript?

JavaScript was developed by Netscape Communications Corporation, a now-defunct software company that was founded in 1994. The initial development of the language was led by Brendan Eich, who was tasked with creating a scripting language that could be used in web browsers.

• What are undeclared and undefined variables?

In JavaScript, undeclared variables and undefined variables are two distinct concepts.

An undeclared variable is a variable that has not been declared using the `var`, `let`, or `const` keywords. Attempting to use an undeclared variable will result in a `ReferenceError`.

• Write the code for adding new elements dynamically?

In JavaScript, you can add new elements to the DOM (Document Object Model) dynamically using the `createElement()` and `appendChild()` methods. Here's an example:

```
// Get a reference to the parent element
let parent = document.getElementById("myDiv");

// Create a new element
let child = document.createElement("p");

// Set the inner text of the new element
child.innerText = "This is a new paragraph.";

// Add the new element to the parent element
parent.appendChild(child);
```

• What is the difference between ViewState and SessionState?

ViewState and SessionState are two different mechanisms for storing data in ASP.NET web applications.

ViewState is used to store the state of a single page between postbacks. It is a hidden field on the page that stores the values of controls and other page-specific data. ViewState is designed to help maintain the state of a page across postbacks, such as when a user clicks a button or performs some other action on the page. ViewState is specific to a single page and is not shared between pages.

SessionState, on the other hand, is used to store data across multiple requests and pages. It is a server-side mechanism that stores user-specific data in a dictionary object that is associated with a session ID. SessionState data is stored on the server, and the client receives only a session ID that is used to retrieve the data from the server.

SessionState data can be shared between pages in the same web application, and can be used to store user-specific data such as login information, shopping cart contents, and other data that needs to persist across multiple pages.

• What is === operator?

The `===` operator in JavaScript is a strict equality operator that compares two values for equality without performing any type conversion. It returns `true` if the values are equal in both value and data type, and `false` otherwise.

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

In JavaScript, you can change the style or class of an element by accessing its properties using the `style` and `classList` objects, respectively.

To change the style of an element, you can use the `style` object to set the value of any of its style properties. For example, to set the background color of an element with the ID "myDiv" to red

• How to read and write a file using JavaScript?

In JavaScript, you can read and write files using the built-in File System module, which provides a way to interact with the file system on the user's machine. However, it's important to note that this module is not available in web browsers, and can only be used in server-side JavaScript environments like Node.js.

To read a file using the File System module, you can use the `readFileSync()` method, which reads the entire contents of a file synchronously and returns it as a string or a buffer.

• What are all the looping structures in JavaScript?

In JavaScript, there are several types of looping structures that allow you to iterate over a set of values or perform a set of actions repeatedly. The main looping structures are:

1. `for` loop: The `for` loop is used to iterate over a set of values for a fixed number of times. It takes three expressions in its header: an initialization expression, a conditional expression, and an update expression.

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

In JavaScript, you can convert a string of any base to an integer using the `parseInt()` method. The method takes two arguments: the string to be converted, and the base of the number system that the string represents.

• What is the function of the delete operator?

In JavaScript, the `delete` operator is used to delete a property from an object or to remove an element from an array.

When the `delete` operator is applied to an array element, it removes the element from the array and shifts any subsequent elements down to fill the gap. However, it does not change the length of the array.

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

n JavaScript, there are three types of popup boxes that can be displayed to the user:

1. Alert Box: An alert box is a simple popup box that displays a message to the user with an OK button. It is typically used to display an important message or to prompt the user for some action. The syntax for displaying an alert box is as follows:

```
alert("This is an alert message");
```

• What is the use of Void (0)?

n JavaScript, `void(0)` is used to evaluate an expression and return `undefined`. The `void` operator is used to specify an expression to be evaluated, and its purpose is to prevent the browser from taking any action when the expression is clicked.

The `void` operator takes an expression as its operand and evaluates it. However, it always returns `undefined`, regardless of the value of the expression. When used with `0` as the operand, it returns `undefined` without causing any side effects.

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

In JavaScript, you can use the `window.location` object to load another page in the browser window. The `location` object provides information about the current URL and allows you to navigate to a new URL.

To load another page, you can use the `location.href` property to set the URL of the new page.

• What are the disadvantages of using innerHTML in JavaScript?

The `innerHTML` property in JavaScript is used to set or get the HTML content of an element. While it can be a convenient way to update the content of a webpage, it has several disadvantages that should be considered:

1. Security Risk: Using `innerHTML` to set HTML content can create a security risk known as "cross-site scripting" (XSS). If the HTML content being inserted contains malicious code, it can be executed by the browser and potentially harm the user's computer.
2. Performance Impact: Setting the `innerHTML` property of an element can be computationally expensive, especially when updating large blocks of HTML. This can slow down the performance of the webpage and lead to a poor user experience.
3. Loss of Event Handlers: When updating the `innerHTML` of an element, any event handlers that were previously attached to the element will be lost. This can cause unexpected behavior and make the webpage harder to debug.
4. Difficulties with Accessibility: The use of `innerHTML` can make it difficult to maintain accessibility of the webpage. Screen readers may have difficulty interpreting the updated HTML content, leading to a poor user experience for visually impaired users.

In summary, while `innerHTML` can be a convenient way to update the content of a webpage, it has several disadvantages that should be carefully considered. Developers should be aware of the potential security risks and performance impact, and should use alternative approaches when possible.