Top 100 JavaScript Interview Questions And Answers

January 6, 2022

JavaScript is, without a doubt, the most famous and widely used programming language in the world. It is also abbreviated as JS but notes that Java and JavaScript are totally different programming languages.

JavaScript is a high-level programming language and easy to learn as well; however, you need a clear understanding of fundamental concepts in order to proceed to become a better programmer.

When you apply for a job, you need to go through an interview where mixed types of questions are asked from basic to advance. Therefore, we have bundled together with the top 100 JavaScript questions that are frequently asked.

Below is the list which contains basic JavaScript interview questions to JavaScript advanced interview questions which will help you clear an interview round. Below JavaScript, interview coding questions are helpful for freshers as well as experienced programmers because we have divided the questions into multiple parts.

Table of Contents

JavaScript Developer Interview Questions

Following are the most often asked JavaScript basics interview questions which will help the newcomers to clear the interviews. The great part about this bundle is, we have also added the HTML, CSS, JavaScript interview questions, JavaScript algorithm interview questions, and so many other particular topic-related questions. Here we go!

Basic JavaScript Interview Questions

Here are some vanilla JavaScript interview questions which will help freshers who don’t have experience in any JS frameworks. Vanilla JavaScript is often called a core JavaScript, don’t be confused!

These JavaScript fundamentals interview questions should be known to every fresher before attending an interview.

Q1. What Is Javascript? What Is The Use Of It?

Ans- This is one of the most basic JavaScript programming interview questions. JavaScript is a client side scripting or programming language which is used to implement complex operations in web pages. JavaScript is present everywhere, presenting timely content updates, interactive maps, dynamic 2D/3D visuals, scrolling video jukeboxes, and so on, you name it.

JavaScript is also used for DOM manipulation, which includes dynamically adding, updating, and deleting HTML elements and CSS properties. Apart from this, JavaScript is also used for desktop application development, and nowadays, we can use JavaScript as a server-side programming language using nodeJS, which is a JavaScript runtime.

Q2. Tell The Actual Name Of Javascript.

Ans- JavaScript was started as Mocha, and then it became a LiveScript. Then it became JavaScript. It is among the most famous and widely used programming languages; however, it has nothing to do with Java programming language. It is said that, because of the popularity of Java, LiveScript became JavaScript to go with Java, with the compiled language.

Q3. Whether Javascript Is Case Sensitive Or Not?

Ans- Yes, JavaScript is case sensitive programming language. This means the keywords, function names, class names, variables, and other identifiers should be written by considering the case sensitivity. For e.g., JavaScript and JavaScript are two different variables, and they will be interpreted differently by JavaScript. Utmost care should be taken while declaring variables and functions.

Q4. Whether Javascript Is Compiled Or Interpreted Language?

Ans- JavaScript is a programming language that is interpreted instead of compiled. Programming languages like C++ or Java are compiled first before running. The original code is run through a compiler, which converts it to bytecode that the machine can understand and execute.

Whereas JavaScript does not require any compilation. Instead, a browser interpreter reads the JavaScript code, interprets each line, and then executes it.

Modern browsers like Google Chrome, Mozilla Firefox, apple safari use just-in-time compilation technology to compile the JavaScript code and convert it to executable bytecode.

Q5. State The Features Of Javascript.

Ans- JavaScript is a programming language that can achieve absolutely anything; website, mobile app, desktop app, server side programs, you guess it. It has some special kinds of features which make complex tasks easier than you think, and that is what makes it a most popular programing language. Here are some amazing features of JavaScript:

- Input validation- Not surprising, but this is among the most useful features of JavaScript. JavaScript is very useful for form validations like validating emails or phone numbers.
- Platform independent- You don't need to install a separate compiler for JavaScript. Almost all browsers support JavaScript, and code runs in a web browser on any platform like Windows, Mac, or Linux.
- DOM manipulation- Using JavaScript, you can have control over the HTML document. You can add, update and delete the HTML elements and CSS properties.
- Object-oriented language- you can have an object-oriented structure rather than functional or monolithic programing.

Q.6 Which Data Types Does Javascript Support?

Ans- In JavaScript, there are six basic data types that can be classified into three groups: primitive, composite, and special data types. The primitive data types are String, Number, and Boolean. Composite data types include Object, Array, and Function (which are all sorts of objects). On the other hand, Null and Undefined are special types of data.

Composite data types can handle collections of values and more complicated entities, whereas primitive data types can only retain one item at a time.

Q7. What Is A Javascript Object?

Ans- In JavaScript, everything is an object. The Object is an entity with properties and types. Or, we can say the Object is a collection of properties, and again, the properties are a set of key-value data. JavaScript object is similar to objects in other programming languages.

We can take an example of a laptop as an object which has multiple properties. A laptop has color, size, battery capacity, storage type, etc. Similarly, JavaScript has an object with properties.

Q8. Difference Between Java And Javascript.

Ans- Very first, Java and JavaScript are totally different programming languages. We will see the difference between these two points by point.

- **Type**– Java is a strongly typed language which means you must declare the type of variables to use throughout the program. Whereas JavaScript is a weakly typed language that means the interpreter automatically detects the type of data as soon as the value is assigned to it.
- **Language type**– Java is an object-oriented programming language, whereas JavaScript is an object-based scripting language.
- **Language scope**– Java is used for both client side and server-side programming and runs in JVM or browser. Whereas JavaScript is a client side scripting language that runs in the browser; however, nowadays, we can

create server side applications also using nodejs.

- **File type**– java code is saved by .java extension, and the class file is generated after code gets successfully compiled. On the other hand, the JavaScript file is saved by the .js extension, and no class file or bytecode is generated for it.

Q9. How Can You Define A Variable In Javascript?

Ans- JavaScript is a weakly typed language, so you don't need to declare the data type of variable while writing it. But, you need to declare the variables using the "var" keyword for e.g., var ab, var arg, etc.

Q.10 What Is Implicit Type Conversion In Javascript?

Ans- In programing language, type conversion is converting data of one type to another. for example, converting a float to int. Implicit type conversion means JavaScript automatically converts the data type of data to another without your acknowledgment. However, implicit type conversion takes place for converting lower data types to higher data types.

Q11. Tell The Procedure To Create An Array In Javascript.

Ans- This is one of the fundamental array interview questions in JavaScript. In JavaScript, you can create an array in two ways. Either traditional method or using "array" keyword. Check syntax below:

- const cars = ["BMW","Audi","Maruti"];
- const cars= new Array("BMW","Audi","Maruti");

Q12. What Is Pass By Reference And Pass By Value?

Ans- The pass-by-value and pass-by-reference points are related to function. It is available in almost every programming language. Pass by reference is passing the address of data/ value to the function where both actual parameter and formal parameters refer to the same memory location. Although, you may use an alias for the passed parameter.

Whereas pass-by-value is the method in which values of actual parameters are passed to the function, but the formal parameters save these values in a separate memory location. The scope of these values is limited to that function only, and any changes made to this value will not reflect the actual parameters.

Q13. Explain Recursion In A Programming Language.

Ans- Recursion is one of the most important parts of any programming language to solve complex problems. Recursion simply means calling the same function again and again until it meets some condition to terminate itself.

Q14. State The Difference Between “==” And “===”.

Ans- “==” and “===” both are comparison operators; however, “==” operator compares two variables irrespective to data type, whereas “===” operator strictly compares two variables and return true only and only if those two variables hold the same value and have the same data type.

Q15. Explain NaN In Javascript With Its Use.

Ans- In JavaScript, NaN stands for Not a Number which simply means the given value is not a valid number. It helps to check whether or not a given value or entered value through the form is a valid number.

> **See also** **Top 100 Ansible Interview Questions And Answers**

Q16. What Is The Use Of The "This" Keyword?

Ans- "this" keyword is present in almost every object-oriented programming language. "this" keyword refers to the current Object or the Object that it belongs to.

In DOM, the "this" keyword is often used to get the properties of the current element as well.

Q17. Explain "BOM" In Javascript. Why Is It Used?

Ans- BOM in JavaScript stands for Browser Object Model. It is different from DOM because instead of interacting with an HTML document, it interacts with the browser. There are a bunch of objects and properties belonging to BOM like a window, screen, history, navigator, location, etc.

Q18. Explain "DOM" In Javascript. What Is It Used For?

Ans- DOM in JavaScript stands for Document Object Model. It is a way to interact with HTML document elements via JavaScript. When the web page is loaded, the browser generates its Document Object Model, and it is constructed as a tree of objects. DOM helps to dynamically change the HTML elements and CSS properties.

Q19. Explain The Difference Between “Let” And “Var” Keywords.

Ans- Let and Var both are used to declare a variable in JavaScript; however, both declare variables in a different way. When you declare a variable using “var”, it will be available throughout the scope. Whereas a variable declared using “let” will be available only for a block. Simply saying, “var” has the scope of function, whereas “let” is a block-scoped.

Q20. Explain The “Typeof” Operator Along With One Example.

Ans- “typeof” operator in JavaScript is exactly the same as “gettype” in PHP. It helps to find the datatype of a JavaScript variable. For example –

```
typeof "JavaScript" //string

typeof 2022 //number

typeof false //boolean

typeof [2,0,2,2] //object
```

Q21. Tell The Difference Between “Undefined” And “Null”.

Ans- Many new learners don’t understand the actual difference between “undefined” and “null” keywords. When you use a variable that is declared but has no value assigned then it is supposed to be an “undefined” whereas “null” is an intentionally assigned value to a variable.

Q22. Explain “IsNaN” In Javascript.

Ans- isNaN function is similar to NaN property. It is a function that returns true if the given value is NaN, i.e., not a number. However, the isNaN function first converts the given value to the number and then tests it.

Q23. Can You Tell The Difference Between Window And Document?

Ans- This is one of the tricky JavaScript objects interview questions that you must be prepared with. When you open a browser, it is a Window that gets loaded first. The Window has some properties like height, width, length, screen, etc.

Talking about the document, it actually gets loaded inside the Window. This document may consist of your scripting code like HTML, PHP, ASPX, etc. To access the Window properties, you use a “window.property” whereas to access the document properties; you may go with something like “window.document.property” which in turn, is available in short as “document.property”.

Q24. Explain JSON And Why Is It Used?

Ans- JSON stands for JavaScript Object Notation which is a lightweight format for data interchange. It is used to send the data between two or more nodes/compters/applications. It consists of human-readable text and is stored as attribute-value pairs and arrays.

JSON is usually used to send and receive the data to and from web applications (data exchange in client and server in both directions). Check detailed information about JSON on Mozilla official JavaScript documentation.

For eg. `{"name" : "JavaScript" , "version" : "ES6"}`

Q25. How Can You Enable Fullscreen Mode Using Javascript?

Ans- JavaScript offers a variety of functions to achieve small to huge operations and tasks. To enable the fullscreen in JavaScript, you have a method “element.requestFullScreen()”. Note that the method name is case-sensitive, and the “element” is an HTML element.

For example, I want to display a video on a full page and it has the id “myvideo”.

Code-

```
var ele= document.getElementById(""myvideo");

ele.requestFullScreen();
```

JavaScript Interview Questions For Experienced Programmers

Q26. Explain What Is Dynamically-Typed Language.

Ans- The typed language is a programming language in which the type of data is known at either compile-time or run time. The language is dynamically typed if the type of variable is known at runtime instead of compile-time. Sometimes, dynamically-typed languages are referred to as loosely-typed languages.

For example, Python, JavaScript is dynamically-typed languages because you don't need to specify the type of variable every time. You are not required to specify that string is a string or a parameter is of type float or something like that.

Q27. What Do You Mean By Asynchronous Programming?

Ans- Asynchronous refers to concurrency. When two or multiple tasks are executing parallel to each other, it is known as asynchronous execution. Asynchronous programming means some task runs independently from the main application and acknowledges the calling thread upon completion, failure, or progress.

It helps to increase the overall performance of the application because no other process depends on the previous process to be completed first. In JavaScript, we can achieve asynchronous programing using concepts like async, await, promises, and callbacks.

Q28. Explain "Immediately Invoked Function" In Javascript.

Ans- In JavaScript, you can define a method or function in multiple ways. You can define a function as an anonymous function, named function, and the functions that are called and executed as soon as they are mounted which are known as "immediately invoked functions".

The name suggests everything to us. We don't need an explicit call to invoke the function. The syntax for creating "immediately invoked function" is as follows:

```
(function(){ //statements})();
```

Q29. What Are “Higher-Order Functions” In Javascript?

Ans- in JavaScript, when a function receives another function as a parameter or returns a new function or both is known as a “higher-order function”. Check the example below:

Code-

```
const msg= function(name){

return function(m){

console.log("hi"+ name);

}

}

const greet= msg("XYZ");

greet("welcome");
```

Q30. Explain Call(), Apply() And Bind() In Javascript.

Ans- call() is a predefined method in JavaScript that is used to invoke a method with the owner object as a parameter. With the call() method, an object can access the method which belongs to another object.

The apply() method in JavaScript is similar to the call() method. Both methods work in a similar way; the only difference between those is that the call() method takes parameters separately, whereas apply() method takes parameters as an array.

The bind() method, unlike the call() and apply() methods, does not instantly run the function. It simply returns a new version of the function with this set to this argument.

Q31. Explain Currying In Javascript.

Ans-Curring is not only limited to JavaScript but it is also used in various other programming languages as well. Currying is a function conversion that converts a function from being callable as f(a, b, c) to being callable as f(a)(b)(c). In simple terms, currying doesn't call a function, it just transforms or converts it. Check the example below.

```
function cur(f) { // curry(f) does the currying transform
return function(x) {
return function(y) {
return f(x,y);
};
};
}
function sum(a,b){ return a+b;}
let cursum= cur(sum);
console.log(cursum(5)(6));
```

Q32. What Is The Scope And Scope Chain In Javascript?

Ans- Scope in JavaScript is related to accessing the variables, functions, and objects in your program. The scope functionality is provided in all the programming to provide the extra layer of security. Basically, there are two types of scope- local scope and global scope.

The local scope of a variable is limited to a block of code, for e.g. a function or a loop. Whereas the global scope of a variable is not limited to the block of code, and once the variable is declared as global, it can be used anywhere in the program.

The scope chain, as the name suggests, is a chain of scopes. Let me explain. When we define a function inside another function, each of those has its own scope. The innermost function has a local scope that is linked to the outer function; this link is called a scope chain.

Q33. What Are Object Prototypes?

Ans- In JavaScript, the technique through which JavaScript objects inherit features from one another is called prototype. JavaScript is also called a prototype-based language that provides inheritance.

It’s possible that an object’s prototype object has a prototype object from which it inherits methods and attributes, and so on. This is known as a prototype chain, and it explains why distinct objects have access to attributes and methods defined on other objects.

Q34. What Is Memoization?

Ans- The terms “memorization” and “memorization” are two totally different concepts in JavaScript. It is very similar to fetching the results from cache. Memoization helps to increase the overall performance of the application by storing the results of expensive functions in the cache and returning them whenever the same inputs are supplied.

Here, expensive function refers to the functions which take a lot of time to execute and consume lots of memory during the execution.

Q35. State Use Of Constructor Function In Javascript.

Ans- Constructors are available in almost every programing language. Constructor is a special method and is used to create and initialize the object of a class. In most programming languages, the constructor has the same name as the class name; however, it totally depends on the programing language.

In JavaScript, the constructor function is used to create and initialize the Object. Like any other programing language, JavaScript also uses the "new" keyword to create an object using the constructor.

Check an example below:

```
function names(){ // constructor function

this.name="john",

this.sirname="wick"

}

const person= new names(); //creating an object
```

Q36. What Is The Arrow Function In Javascript?

Ans- The array function in the JavaScript was introduced in the ES6 version and it is supposed as one of the most popular features in ES6. It allows you to write a shorthand version of a function definition.

The previous method to create a function-

```
fun= function(a,b){

console.log("hello")

}

Using arrow function-

fun=()=>{

console.log("hello")

}
```

This syntax further can get shorter. If the function body contains only one statement that returns something, then you can write a function something like-

```
fun=()=> "hello"
```

You can use the parameters to the function as well.

```
fun=(val)=>"hello"+val;
```

Q37. Explain The Use Of The Array Slice Method.

Ans- It is one of the most important JavaScript array interview questions. There are a few methods in JavaScript with almost identical names, and students get confused between them, but you shouldn't!

The array slice method is used to return the set of elements from the array as specified in the parameters. It doesn't change the original array and returns up to (last-1) elements.

Note- The array index starts from 0.

Check the below example.

```
var names=["john", "naruto", "saitama", "gojo"];

console.log(names.slice(1,3);

//output-naruto, saitama
```

Here, we have specified parameters 1 and 3, which means we want to get elements starting from index 1 to (3-1), i.e., 2. That means we should get the second and third elements in the array.

Q38. Difference Between Object And Map.

Ans- In JavaScript, both map and object store values in key-value pairs. But there are slight differences between these two. The map is a data structure that contains the unique key to which the value is mapped. The keys are always unique in the map data structure so it avoids data duplication.

In Object, keys are generally specified as property and are associated with a single value.

The main difference between objects and maps is, keys in objects are of type strings or int, whereas, in the map, keys can be of any type.

The way to create objects and maps is also different. You don't need to use any built-in function or method to create an object, whereas to create a map, you must use a "map" constructor.

Eg. create an object-

```
const obj={

name:"john",

"age": 50

}

create a map-

const map= new map([

["name","john"],

["age",50]

]);
```

Q39. What Is The First-Class Function?

Ans- In JavaScript, we can use function as a variable and store the value of the function in a variable. The JavaScript treats function like any other variable or an object that means you can store them in a variable, pass them as arguments, return from the function, and so on.

Q40. How To Redeclare A Variable In The Switch Block Without Any Error?

Ans- If you initialize the variable in the switch case, it gets the scope of the whole block without even knowing yourself. To avoid this, you just need to put curly braces around the particular case block. Check the code below.

switch (a)

```
{
case 0:
  int b = 5;
  cout << "I declared variable b." << end;
  break;
case 1:
  cout << "variable b has scope here too!" << end;
  break;
default:
  break;
}
```

The above code will generate a compile-time error. To solve this, you can declare a variable inside of curly braces, so it will be declared as a blocked scope variable. Check the right code below.

switch (a)

```
{
case 0:
  {
    int b = 5;
    cout << "I declared variable b." << end;
  }
  break;
case 1:
  cout << "I don't have access to b." << end;
  break;
default:

  break;

}
```

Q41. What Is A Module? What Is The Benefit Of Using It?

Ans- Module is similar to the library, which consists of reusable code that is intended to be used as the building block in the JavaScript application. Modules are small independent units of code that let the developers define private and public data members and member functions.

In JavaScript, you can have your own created module or a third-party module as well, that you can install and use in your application via package managers like NPM. The main benefit of creating and using modules is the reusability and the app’s performance.

Q42. How To Manipulate Dom Using A Service Worker?

Ans- Before seeing how to manipulate DOM using a service worker, let’s see what is the service worker. A service worker is a script that runs in the background of your browser, independent of a web page, allowing you to access services that don’t require a web page or user involvement.

The service worker is a JavaScript worker, so it cannot directly access and update the DOM. So instead of directly manipulating the DOM, a service worker communicates with the pages that it controls by giving a response to a message that is sent via the postMessage interface, and then those pages can manipulate the DOM.

Q43. What Is A Cookie? Why Do We Need It?

Ans- You can think of a cookie as data that is stored on your local machine as a text file. A cookie helps the website to remember the information about the user, which then helps to visit the site again. In general, it stores the information of a visited user and uses that information to make the site more useful to that user.

See also **Top 100 Ansible Interview Questions And Answers**

For example, you can store the id or username in the cookie and set the time limit to destroy itself so that the next time user opens the site within the specified time limit; the user will be logged in automatically without entering id and passwords.

Another use case of a cookie is, if your website is more like a web application like a todo app or a note app, you can store the last visited URL in a cookie so that whenever the user closes the browser and opens it again after some time, the web application will be redirected to that specific last visited URL which reminds the user what was he performing during the last session.

Q44. Difference Between A Cookie, LocalStorage, And SessionStorage.

Ans- Cookie, localStorage, and sessionStorage, are all used to store some information. We will see each of those, one by one, and you will understand the difference between them easily.

A cookie is a small text file that stores and remembers the user's data on the local machine. It holds the data in key-value pairs. You can set the expiry date/time to the cookies; if not specified, cookies will be deleted on browser close.

The localStorage and sessionStorage are relatively new features of JavaScript and not necessarily be supported by legacy browsers. The localStorage will store the data on your local machine and has no expiration time. The localStorage can store the data up to 5 MB using JavaScript and store the data as plain text.

The sessionStorage is quite similar to the localStorage, but as the name suggests, it only lasts for the session, meaning that all the data stored in sessionStorage will be wiped out on browser close.

Check out this **StackOverflow thread** to know more in-depth.

Q45. What Is The Primary Difference Between Local Storage And Session Storage?

Ans- The localStorage and sessionStorage are both used to store some information on the client-side. However, there is a slight difference between them. The localStorage stores the data in key-value pairs which have no expiration date.

This means, even if you close the browser and open it again on next day, the data/information in localStorage remains as it is. However, the user can delete all the localStorage data from the browser.

On the other hand, sessionStorage stores the data in key-value pair same as localStorage but have life until the browser close. As soon as the user closes the browser, sessionStorage information is cleared.

Q46. How To Check The Web Worker’s Browser Support In Javascript?

Ans- There are two ways you can check if the browser supports web workers using JavaScript.

code-

```
tests[webWorkers]= function(){

return !!window.Worker;

}

By another way,

code- if(typeof(Worker)!=="undefined"){

//broser supports web workers

}else{

//sorry, browser doesn't support web workers

}
```

Q47. Explain Strict Mode In Javascript.

Ans- Strict mode feature is introduced in the ECMAScript version 5. The strict mode in JavaScript is achieved by the “use strict” literal expression. The strict mode won’t allow you to use undeclared variables. The main purpose of introducing strict mode in JavaScript is to enhance security.

In an earlier version of JS, if the programmer mistypes a variable name, it creates a new global-scope variable. Using strict mode, there is no possibility of accidentally creating new variables.

The syntax of strict mode is=

code – `"use strict";`

Q49. What Is The Use Of Double Exclamation?

Ans- In JavaScript, !! Converts object to the boolean. If you closely look at it, you will realize it is two times “NOT”. Let me explain to you. The first “!” is NOT, and again “!” is the second NOT, which means you are first converting the value to boolean, inverting it, and again inverting it.

See, ! is the “NOT”; so, !true is false and !false is true. In terms on 0 and 1, !0 is 1 (true) and !1 is 0(false).

If the value is falsey(0, null, undefined), it will be false else true.

For example-

console.log(!!0) // returns false because 0 is falsey value

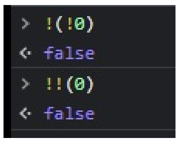
console.log(!!1) // returns true because its truthy value

Explainationsimplify the expression first-

ex 1- !!0 is equivalent to !(!0)

!(!0) = !(1) = (0)

so !!0 is 0 = false.

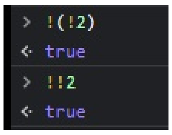


```
> !(!0)
< false
> !!(0)
< false
```

ex 2- !!1 equivalent to !(!1)

!(!1)= !(0) = (1)

so !!1 is 1 = true



```
> !(!2)
< true
> !!2
< true
```

Q50. Explain The Use Of The Delete Operator.

Ans- The JavaScript delete operator is used to remove the object property.

Syntax- delete expression;

eg. code=

```
var names= {

first:"john",

last:"wick"

}

console.log(names); //show first and last names

delete names.first;

console.log(names); // only show last name
```

Q51. How Can You Access History In Javascript?

Ans- In JavaScript, history is a part of BOM, or you can say it is an object of BOM. Whenever you navigate to the different pages or open a new web page, a record is created in history. To manipulate the browser history, you get a "history" object.

Syntax of history object-

window.history;

To go backward-

window.history.back() or history.back()

To go forward-

window.history.forward() or history.forward();

To navigate using steps-

window.go(2); //to go forward

window.go(-2); //to go backward

Q52. How To Detect Caps Lock Is On Or Off In Javascript?

Ans- Capslock is a key on the keyboard so we have to use a keyboard event listener to detect if the caps lock is active. JavaScript provides us getModifierState() method of a keyboardEvent object to return the status of the modifier. If the modifier is active, it returns true else false. Check the example below.

code-

```
if(e.getModifierState('capslock')){

//show message on label capslock is on

}else{

//capslock is disabled

}
```

Where e is the event object and you can get it from the keyup or keypressed event. check code below:

code-

```
var ip= document.getElementById("inputField");

ip.addEventListener("keyup", function(event){

if(event.getModifierState('capslock')){

alert("caps lock is on");

}else{

}

});
```

Q53. What Is Event Bubbling In Javascript?

Ans-Event bubbling is a way of event propagation in the HTML DOM API. When an event is in an element inside another element and both elements have registered a handle to that event, event bubbling happens.

It's a procedure that starts with the element that caused the event and then moves up the hierarchy to the contained elements. The event is caught and processed by the innermost element first, and then transmitted to the outermost elements through event bubbling.

Q54. How To Find Os Details Using Javascript?

Ans- In JavaScript, you can find the operating system details using two ways; using either navigator.appVersion property or navigator.userAgent property. Here is the code that displays the name of the operating system.

Save the following code by .HTML extension and open it in the browser to see result.

code-

```
<!DOCTYPE HTML>

<html>

<head>

  <title> JavaScript Detecting the Operating System of User. </title>

</head>

<body style="text-align:center;" id="body">

  <h1 style="color:green;">   JavaScript <3  </h1>

  <p id="UP" style="font-size: 19px; font-weight: bold;"> </p>

  <button onclick="Fun()">  click here </button>

  <p id="DOWN"

  style="color: green; font-size: 24px; font-weight: bold;"> </p>

  <script>

    HTMLDocument.prototype.e = document.getElementById;

    var el_up = document.e("UP");

    var el_down = document.e("DOWN");

    el_up.innerHTML = "Click on the button to get the OS of User's System.";

    var Name = "Not known";

    if (navigator.appVersion.indexOf("Win") != -1) Name =

    "Windows OS";

    if (navigator.appVersion.indexOf("Mac") != -1) Name =

    "MacOS";

    if (navigator.appVersion.indexOf("X11") != -1) Name =

    "UNIX OS";

    if (navigator.appVersion.indexOf("Linux") != -1) Name =

    "Linux OS";


    function Fun() {

      el_down.innerHTML = Name;
    }
  </script>
</body>
</html>
```

Q55. Difference Between Native Objects And Host Objects.

Ans- The native objects and host objects specification is available in the ECMAScript 2015. Host Items are objects that come from a certain context. They aren't always the same since each environment is unique and has host objects that enable ECMAScript execution. The browser environment, for example, provides things such as windows. Objects like NodeList are available in a node.js/server context.

On the other hand, Native Objects, also known as Built-in Objects, are JavaScript's standard built-in objects. Native objects are also known as 'Global Things' because they are objects that JavaScript has made available for use natively. There are a number of native objects available, which you can check on **Mozilla's official website**.

Q56. How Can You Debug The Javascript Code?

Ans- There are multiple ways to debug the JavaScript code. You can use the browsers' debugging environment which is common for most developers out there. All the major browsers like Google Chrome, Mozilla Firefox, Apple Safari, Microsoft Edge, etc. provide debugging tools.

Apart from this, you can have the following ways to debug the JavaScript code.

- console.log() method to check whether a particular statement throws the error.
- You can set breakpoints in the browser's debugger tool.
- Using the "debugger" keyword in JavaScript, you can easily debug the code.

Q57. What Is A Callback In Javascript?

Ans- As the name suggests, call the function after the particular work completes. In simple terms, the callback in JavaScript is a function that is passed to another function as a parameter to be executed later. The callback function executes when another function completes its execution. For example, when you have to perform operations on the numbers after the addition is done. Check the example below.

Code-

```
function display(sum){

console.log(sum/10); //displays the division

}

function addition(a,b,callback){

let sum= a+b;

callback(sum); //calls the display function

}

addition(10,10,display); //function call with the display function as a parameter
```

Q58. What Do You Mean By Escape Characters In Javascript?

Ans-The symbol used to start an escape command in order to perform some activity is known as an escape character. Escape characters are present in almost every programming language and all of them are nearly identical.

In JavaScript, there are a bunch of escape characters present that you can use throughout the program. Following are the available escape characters in JavaScript-

- \n = new line
- \b = backspace
- \\ = backslash
- \” = double quote
- \’ = single quote
- \t = horizontal tabulator
- \v = vertical tabulator

Q59. How Can You Use An External Javascript File?

Ans- External JavaScript files are useful for code reusability, which simply means you can write the piece of code in a separate file that you need multiple times in your application and include it in the main application file.

In the HTML file, you can include the js file in following way:

```
<script type="text/JavaScript" src="jsfile.js"> </script>
```

Q60. Tell The Purpose Of The Window Object.

Ans- In JavaScript, the Window is a tab that is opened in front of you. The window object has document property which consists of the DOM document and the nodes and methods which we can use for DOM manipulation.

When you create any variable without the let, const, or var keywords, it automatically becomes a part of a window object. In a web browser, the window object refers to the opened tab and there can be any number of window objects depending on the number of tabs opened.

Q61. What's The Difference Between Event.Preventdefault() And Event.Stoppropagation() Methods In Javascript?

Ans- The two methods, event.preventDefault() and event.stopPropagation() are used to prevent something, but what? The preventDefault() method prevents the browsers' default action like clicking on the link redirecting you to a new page.

On the other hand, the stopPropagation() method prevents further propagation in the capturing and bubbling phase of the current event.

Q62. How To Convert String To Integer In Javascript?

Ans- Converting a value from one type to another type is tricky and many programming languages offer prebuilt functions to do so. JavaScript offers you an easy way to convert a string to an integer using the method called "parseInt".

The syntax to convert a string to an integer is:

parseInt(string);

eg.

- `parseInt("10") //returns 10`
- `parseInt(" 10 ") // returns 10`
- `parseInt("10 ok") // returns 10`
- `parseInt("ok 10"); // returns NaN`

Key points to remember before using parseInt method in JavaScript-

- It ignores trailing and leading spaces
- Only the First integer in the string found is returned
- The first character is parsed first to the int and converted; returns NaN if fails

Q63. Explain Import And Export Keywords In Javascript.

Ans- As the names suggest, these keywords are used to import and export the data members and member functions from and to different js files.

The import and export keywords were introduced in the ES6 version. Using the ES6 standards, we can create modules in the JavaScript, which can be used in other js files for reusability. These modules contain classes, variables, and functions.

The export keyword is used to export a variable_name from the current file.

The import keyword is used to import a variable_name from another file that exports that variable.

The variable_name is anything like a function, class, or whole js file.

Q64. Difference Between View State And Session State.

Ans- In JavaScript, ViewState is used for client-side state management, whereas sessionState is used for server-side state management.

In ViewState, changes made on one page don't reflect on other pages. Because it is client-side state management, data gathered using ViewState is stored for the client and cannot be sent across.

On the other hand, sessionState stores and maintains the data on the server-side. The information that is gathered by sessionState remains as long as the user is active and gets destroyed when the session is completed, or the user closes the browser.

Q65. Is Javascript Faster Than Asp? Justify.

Ans- JavaScript is faster than ASP because it is a client-side language, whereas ASP is a server-side language. JavaScript doesn't need to communicate with the server as it is made for client-side scripting purposes, but ASP needs to communicate with the server in order to exchange information.

However, the NodeJS, which is a run time of a JavaScript that is made for server-side scripting, is comparatively slower than ASP.

Q66. Explain The Use Of The Debugger Keyword.

Ans- debugger keyword in JavaScript is the same as a breakpoint used in the various debugger tools. The debugger keyword stops the execution before the next instruction executes in the browser.

Copy and paste the following code into an HTML file and run it in the browser to check the working of the debugger keyword.

```
<!DOCTYPE HTML>

<html>

<head>

  <title>JavaScript debugger</title>

</head>

<body style="text-align:center;" id="body">

  <h4> demonstrations of Debugger keyword</h4>

  The solution of 10 * 20 is:

  <p id="test"></p>

  <p>code stopsexecution at the start of the debugger If the debugger is turned on</p>

  <script>

    debugger;

    var x = 20;

    document.getElementById("test").innerHTML = x;

    debugger;

    var y = 5;

    document.getElementById("test").innerHTML = y;

    var z = x * y;

    debugger;

    document.getElementById("test").innerHTML = z;

  </script>

</body>

</html>
```

Javascript Closure Interview Questions

JavaScript closure, hoisting, callbacks, and promises are some of the important topics for advance level development. There is no doubt you will face half of the questions based on these topics if you are an experienced programmer.

See also **Top 100 Ansible Interview Questions And Answers**

Q67. What Is Closure In Javascript? What Is It Used For?

Ans- Closure is one of the important parts of the modern JavaScript paradigm. A closure is a function containing references to its surroundings that have been bundled together (enclosed) (the lexical environment).

Simply put, a closure enables an inner function to get access to the scope of an external function. When a function is generated in JavaScript, closures are formed throughout the function creation process.

Q68. Tell Me The Advantages Of Closure.

Ans- The closure allows to bind a variable to an execution context in the JavaScript. In simple terms, the main purpose of the closure is data encapsulation.

Because most APIs that require callback functions (for example, an “onclick” function) do not provide other means to transmit parameters to those callback functions, closures are required in JavaScript (or to explicitly set the “this” pointer). To allow the callback to access variables in the “parent” function, you must use closures.

Q69. State The Difference Between Closure And Scope.

Ans- The closure and scope can be explained in simple terms. The variable is scoped to the particular function if it is declared inside the function and can’t be used outside it.

On the other hand, when you define a function inside of another function then the inner function is called a closure.

Q70. What Is Promise In Javascript?

Ans- As the name suggests, promise in JavaScript assures you a result. In any code, there is two following- a producing code and a consuming code. A producing code is something that performs something and takes time. On the other hand, a consuming code is something that takes results from the producing code.

A promise in JavaScript is an object that connects producing code and consuming code. As soon as producing code gets the result, it must call the consuming code; either the output is successful or generates an error.

Javascript Promises Interview Questions

Q71. What Are The Three States Of Promise?

Ans- In JavaScript, a promise can be pending, fulfilled, or rejected. There are two properties of a promise; state and result. When the promise is pending, the result is undefined. When the promise is fulfilled, you get value as a result. When the promise is rejected, you get an error object as a result.

Q72. State Benefits And Drawbacks Of Promises Over Callbacks.

Ans- There are advantages and disadvantages as well of using promises in JavaScript over callbacks.

The advantages are:

- Good control flow of asynchronous logic.
- Code maintainability
- Reduced coupling

The disadvantages are:

- Kills the intention of asynchronous programming
- One Object is returned only
- multiple arguments cannot be returned

Q73. How To Write Sleep Function Using Promise?

Ans- Following code is used to write sleep function using promise in JavaScript.

code-

```
function sl(timeMS) {
    return new Promise(result => {
      setTimeout(result, timeMs);
    });
  }
```

Javascript Hoisting Interview Questions

Q74. What Is Hoisting? What Is The Use Of It?

Ans-Hoisting is the default behavior in JavaScript, which moves all declarations to the top of the scope before code execution. Essentially, it allows us to benefit from the fact that no matter where functions and variables are declared, they are moved to the top of their scope, whether global or local.

Q75. How Many Ways Can You Create Function Hoisting In Javascript?

Ans- In JavaScript, you can create a function in two ways, through function declaration and function expression. The function declaration hoists the fiction definition in JavaScript. On the other hand, function expressions are not hoisted in JavaScript. Hence, there is no way to use function expressions before you define them.

HTML CSS JavaScript Interview Questions

Since JavaScript is a client-side programing language, you may be asked some frontend questions as well. Hence, we have found a bunch of JavaScript front end interview questions here if you are giving an interview for frontend developer role.

These are some entry level JavaScript interview questions which include basic HTML and CSS questions as well.

Q76. What Is The Role Of Javascript In Client-Side Programming?

Ans- JavaScript is a client-side programming/scripting language and is the backbone of the webpage to make it interactive. In short, it is responsible for making a webpage alive.

Using JavaScript for the web, you can perform multiple things like DOM manipulation, event handling, and so on. JavaScript is not limited to the web; you can use it for mobile application development as well.

Q77. In How Many Ways Can You Validate Email In Javascript?

Ans- There are two ways you can validate an email in JavaScript; using either a prebuilt feature or a regex (regular expression). The HTML allows you to set the type of input field on the form so that it will automatically understand if the entered value is valid or not. Some of the prebuilt types in HTML forms are email, number, text, radio, checkbox, submit, etc.

However, the email type of input field is not a strict validator for an email. If you want to check an email as per the standards, you can use the following regular expression.

regular expression for email validation- /^(([^<>()[\]\\.,;:\s@”]+(\.[^<>()[\]\\.,;:\s@”]+)*)|(“.+”))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/

Here is the JavaScript code to perform email validation-

```
const validate= (email) => {

 return String(email).toLowerCase().match(

/^(([^<>()[\]\\.,;:\s@"]+(\.[^<>()[\]\\.,;:\s@"]+)*)|(".+"))@((\[[0-9]{1,3}\.[0-9]{1,3}\.[0-9]
{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/

  );};
```

Q78. How Can You Change The Html Document Background Color Using Javascript?

Ans- To change the HTML document background color, you can use the DOM manipulation property of the JavaScript. Following is the code to change the background color of a body of an HTML page when the user clicks on a button.

```
<!DOCTYPE HTML>

<html>

<head>

  <title>JavaScript change background color</title>

</head>

<body style="text-align:center;" id="body">

  <button id="btn">change body background color</button>

  <script>

    document.getElementById("btn").addEventListener("click",function(){

      document.getElementById("body").style.backgroundColor="red";

    })

  </script>

</body>

</html>
```

Q79. How Can You Submit A Form Using Javascript?

Ans- You can submit a form using JavaScript by clicking on a button. You can add an event listener to the button and perform a submit action on it. Check the example below:

```
<!DOCTYPE HTML>

<html>

<head>

  <title>JavaScript submit form using JavaScript</title>

</head>

<body style="text-align:center;" id="body">

  <form action="" id="form">

    <label for="">enter name</label>

    <input type="text" name="name">

    <br>

    <input type="button" onclick="submitForm()" value="submit">

  </form>

  <script>

    function submitForm(){

    document.getElementById("form").submit();

    }

  </script>

</body>

</html>
```

Q80. Explain How To Submit A Form On Link Click In Javascript.

Ans- You can submit a form by clicking on a link as well in the JavaScript. Check the code below-

```
<!DOCTYPE HTML>

<html>

<head>

  <title>JavaScript submit form on link click</title>

</head>

<body style="text-align:center;" id="body">

  <form action="" id="form">

    <label for="">enter name</label>

    <input type="text" name="name">

    <br>

    <a href="#" onclick="submitForm()">submit Form</a>

  </form>

  <script>

    function submitForm(){

    document.getElementById("form").submit();

    }

  </script>

</body>

</html>
```

Q81. State Difference Between Innerhtml And Innertext.

Ans- The innerHTML returns everything that a block of HTML element consists including other HTML tags. On the other hand, innerText only returns the plain text. Check the below example to understand it better.

code-

```
<!DOCTYPE html>

<html>

 <head>

  <title>innerText vs innerHTML</title>

 </head>
```

```
   <body style="text-align: center" id="body">
    <div id="div">
     <h3>h3 tag</h3>
     <p>another p tag</p>
    </div>
    <button id="btn1" onclick="innnerHTML()">innerHTML</button>
    <button id="btn2" onclick="innnerText()">innerText</button>
    <script>
     function innnerHTML() {
      alert(document.getElementById("div").innerHTML);
     }
     function innnerText() {
      alert(document.getElementById("div").innerText);
     }
    </script>
   </body>
  </html>
```

innerHTML-

innerText-

Q82. Name A Few Javascript Frontend Libraries And Frameworks.

Ans- There are thousands of JavaScript libraries and frameworks available out there. However, here are among the famous in the JavaScript community as per the usage, simplicity, flexibility, and features. These libraries and frameworks are created to serve different purposes like charts, images, DOM manipulation, games, forms, animation, math functions, UI, etc.

- jQuery- used for Easy DOM manipulation, JSON parsing, AJAX, etc.
- ReactJS- Frontend library for creating interactive single-page website/applications (SPA).
- AngularJS- A JS framework to create modern and dynamic single-page applications.
- AnimeJS- Popular JS library for animations.
- AOS- Library to add effects to element on scroll.
- ChartJS- Most popular library to create charts in your website/web application.

Q83. What Is The Difference Between Client-Side Javascript And Server Side Javascript?

Ans- The client-side JavaScript is used to perform actions on the user side and doesn't exchange data with the server. JavaScript is basically developed for client-side programming.

However, NodeJS, a JavaScript runtime, is developed to use JavaScript as a server-side programming language. Using server-side JavaScript, you can interact with the database, exchange data from and to the client's machine and server.

Q84. In How Many Ways, You Can Access An Html Element Using Javascript?

Ans- Using DOM manipulation in JavaScript, you can access an HTML element in multiple ways. Here are the ways to access an HTML element with the syntax-

- get element by ID- document.getElementById(id);
- get element by class- document.getElementsByClassName(class);
- get element by name- document.getElementsByName(name);
- get element by tag name – document.getElementsByTagName(name);

Q85. Explain Load And Domcontentloaded In Javascript.

Ans- The load and DOMContentLoaded are two events in JavaScript that are used to check if the webpage is loaded completely. However, both work in a slightly different way.

The DOMContentLoaded gets executed as soon as the basic HTML structure of the page gets loaded, which only consists of HTML tags and excludes CSS, js files, and images.

On the other hand, the load event gets executed as soon as the whole HTML page including the CSS, js files, and images are loaded.

Q86. In How Many Ways, Javascript Code Can Be Involved In Html Files?

Ans-You can include the JavaScript code in two ways in your HTML document, either by writing a code inside a script tag or adding an external js file.

For e.g.,

Method 1-(code)-

```
<body>

<h1>heading</h1>

<script>

//js code

</script>

</body>
```

method 2- (code)

```
<body>

<h1>heading</h1>

<script type="text/JavaScript" src="externalJSFile.js"></script>

</body>
```

JavaScript Algorithms Interview Questions

Q87. Check Number Is Prime Or Not.

Ans-

```
function checkPrime(n) {

  var dvsr = 2; //start divisor from 2

  while (n > dvsr) {

    if (n % dvsr == 0) {

      return false;

    }

    else

      dvsr++; //increment division

  }

  return true;

}

console.log(checkPrime(19));
```

Q88. Find The Nth Fibonacci Term.

Ans-

```
function fib(n) {

  var fibe = [0, 1];

  if (n <= 2) {

    return 1;

  }

  for (var i = 2; i <= n; i++) {

    fibe[i] = fibe[i - 2] + fibe[i - 1];

  }
  return fibe[n];
}
console.log(fib(4));
```

Q89. Swap Two Numbers.

Ans-

```
function swap(a, b) {
  console.log(`before swap=a=${a} , b= ${b}`);
  var t;
  t = a;
  a = b;
  b = t;
  console.log(`after swap=a=${a} , b= ${b}`);
}
swap(10,20);
```

Q90. Swap Two Numbers Without A Third Variable.

Ans –

```
console.log(`before swap=a=${a} , b= ${b}`);
  b = b - a;
  a = a + b;
  b = a - b;
  console.log(`after swap=a=${a} , b= ${b}`);
}
swap(10, 20);
```

Q91. Reverse The String

Ans-

```
function reverseString(string){
  var reverse = '';
  for(var i = string.length-1; i>=0;i--){
   reverse +=string[i];
  }
  console.log(reverse);
 }
```

```
reverseString("reverse string");
```

Q91. Check Whether The Word Is Palindrome Or Not.

Ans-

```
function checkPalindrome(string) {
  var i, length = string.length;
  for (i = 0; i < length / 2; i++) {
    if (string[i] !== string[length - 1 - i])
      return false;
  }
  return true;
}
console.log(checkPalindrome("tut")); //true
console.log(checkPalindrome("tuts")); // false
```

JavaScript Es6 Interview Questions

(https://www.javatpoint.com/es6-interview-questions)

If you want to make a career as a JavaScript developer then you must be familiar with ECMAScript or ES. It is a superset of JavaScript and releases new features and upgrades. The major release of ECMAScript is ES6 which was released in 2015; therefore, it is often called ECMAScript 2015 as well.

Q92. Explain Ecmascript. How Is It Related To Javascript?

Ans- You can think of an ECMAScript as a parent that creates a set of rules for the scripting languages such as JavaScript, script, etc. JavaScript is based on the rules of ECMAScript standards.

Q93. State New Features Introduced In Es6.

Ans- The ES6 was the major release for JavaScript that includes a bunch of new features which helped programmers to overcome various difficulties. Following are the features that were introduced in ES6.

- arrows
- classes
- object literals
- template strings
- destructuring
- let and const keywords
- Unicode
- modules

- map, set, weakmap, weakset
- proxies
- symbol
- math, number, string, array, and Object APIs
- tail calls
- reflect API

Q94. Explain Let And Const Keywords.

Ans- In JavaScript, you can decide the scope of the variable by the way how you declare it. There are three ways you can declare a variable in JavaScript; using var, let, and const.

Let and const keywords are used to create a block-scope variable. The let and const keywords are very similar to each other. The only difference between them is that you cannot reassign the value to the variable which is declared as const.

Code-

```
let a=10;

a=20;

console.log(a); //returns 20

const b=20;

b=30;

console.log(b); //returns error
```

Q95. What Are Template Literals?

Ans- Template literals, also known as template strings, is one of the best features introduced in version ES6. Alternatively, you can call it a string template or a back-tics syntax.

Using template literals, you can create a string that starts and ends with the backticks (`) which is capable of consisting of expressions. It helps you to write double-quoted strings inside it or display variables and concatenate two strings easily as well.

for example-

```
var a=10, b=20, str="string";

console.log(`values of a=${a}, b= ${b} , str= ${str}`);

//output- values of a=10, b= 20 , str= string
```

Output-

Q96. Explain Destructuring Assignment.

Ans- As the name suggests, restructuring allows you to break down the arrays, objects into separate values. It just unpacks the value from an array or an object and does not destruct or modify the existing array/object.

Example-

```
class cl {

constructor(){

console.log("constructor");

}

  fun() {

    console.log("JavaScript method inside class");

  }

};

var c = new cl(); //creating object c of class cl

c.fun(); //calling a method

//output-

constructor

JavaScript method inside class
```

Q98. What Is The Generator Function?

Ans- A generator function is defined as functionName* that returns a Generator object in JavaScript. A generator function is a function that can be exited and later entered again. The Generators in JavaScript are more powerful when used with promises in asynchronous programming.

Q99. Explain Default Parameters.

Ans- As the name suggests, default parameters are the default values when no parameter is passed. Check the example below –

```
function mul(a, b = 10) {

  console.log(a * b);

}

mul(10, 20); //output -200

mul(10); //output - 100
```

Q100.Define Babel.

Ans- Writing a JavaScript code that runs in the older version of the JavaScript engine is a kind of headache. That is where Babel comes into action. Babel is a transcompiler that converts ES6+ code into a backward-compatible version of JavaScript which can run in older JS engines.

Conclusion

As stated earlier, JavaScript is one of the most popular programming languages and in high demand. To get a high-paying job, these JavaScript interview questions will help you to clear the coding rounds as well as interview rounds.

It is recommended to try these questions and coding answers yourself before attending any interview so that you will have a rough experience about how the particular function or method works and you won't get confused when you are asked any question. Happy coding!

Interview Questions
‹ 15 Best Free Photo Editor Software For Windows PC
› Top 30+ Best Internet Of Things Companies In The World