Home Resources Software Development JavaScript Tutorial: Learn JavaScript from Scratch 90+ JavaScript Interview Questions and Answers [2025]

90+ JavaScript Interview Questions and Answers [2025]

Lesson 29 of 36

By Haroon Ahamed Kitthu

Last updated on May 9, 2025

625776

Table of Contents

View More

JavaScript is an open-source programming language designed for creating web-centric applications. It is lightweight and interpreted, which makes it much faster than other languages. JavaScript is integrated with HTML, which makes it easier to implement in web applications.

This article provides you with a comprehensive list of common JavaScript interview questions and answers that often come up in interviews. It will also help you understand the fundam concepts of JavaScript.

JavaScript Interview Questions for Freshers

Here are some basic JavaScript interview questions and answers for you to prepare during your interviews.

1. What do you understand about JavaScript?

JavaScript is a popular web scripting language used for client-side and server-side development. Its code can be inserted into HTML pages and understood and executed by web browsers. JavaScript also supports object-oriented programming abilities.

2. What's the difference between JavaScript and Java?

JavaScript	Java
JavaScript is an object-oriented scripting language.	Java is an object-oriented programming language.
JavaScript applications are meant to run inside a web browser.	Java applications are generally made for use in operating systems and virtual machine

JavaScript does not need compilation before running the application code.	Java source code needs a compiler before it can run in real time.

3. What are the various data types that exist in JavaScript?

These are the different types of data that JavaScript supports:

- Boolean - For true and false values
- Null - For empty or unknown values
- Undefined - For variables that are only declared and not defined or initialized
- Number - For integer and floating-point numbers
- String - For characters and alphanumeric values
- Object - For collections or complex values
- Symbols - For unique identifiers for objects

4. What are the features of JavaScript?

These are the features of JavaScript:

- Lightweight, interpreted programming language
- Cross-platform compatible
- Open-source
- Object-oriented
- Integration with other backend and frontend technologies
- Used especially for the development of network-based applications

}

5. What are the advantages of JavaScript over other web technologies?

These are the advantages of JavaScript:

- Enhanced Interaction

JavaScript adds interaction to otherwise static web pages and makes them react to users' inputs.

- Quick Feedback

There is no need for a web page to reload when running JavaScript. For example, form input validation.

- Rich User Interface

JavaScript helps make the UI of web applications look and feel much better.

- Frameworks

JavaScript has countless frameworks and libraries that are extensively used for developing web applications and games.

6. How do you create an object in JavaScript?

Since JavaScript is essentially an object-oriented scripting language, it supports and encourages the usage of objects while developing web applications.

```
const student = {

  name: 'John',

  age: 17

}
```

7. How do you create an array in JavaScript?

Here is a straightforward way of creating arrays in JavaScript using the array literal:

var a = [];

var b = [‘a’, ‘b’, ‘c’, ‘d’, ‘e’];

8. What are some of the built-in methods in JavaScript?

Built-in Method	Values
Date()	Returns the present date and time
concat()	Joins two strings and returns the new string
push()	Adds an item to an array
pop()	Removes and also returns the last element of an array

round()	Rounds of the value to the nearest integer and then returns it
length()	Returns the length of a string

9. What are the scopes of a variable in JavaScript?

The scope of a variable implies where the variable has been declared or defined in a JavaScript program. There are two scopes of a variable:

- Global Scope

Global variables with global scope are available everywhere in JavaScript code.

- Local Scope

Local variables are accessible only within the function in which they are defined.

10. What is the ‘this’ keyword in JavaScript?

The ‘this’ keyword in JavaScript refers to the currently calling object. It is commonly used in constructors to assign values to object properties.

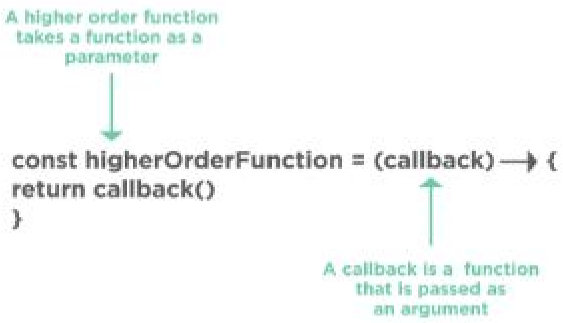
11. What are the conventions of naming a variable in JavaScript?

Following are the naming conventions for a variable in JavaScript:

- Variable names cannot be similar to that of reserved keywords. For example, var, let, const, etc.
- Variable names cannot begin with a numeric value. They must only begin with a letter or an underscore character.
- Variable names are case-sensitive.

12. What is Callback in JavaScript?

In JavaScript, functions are objects; therefore, functions can take other functions as arguments and be returned by other functions.


A higher order function takes a function as a parameter
A callback is a function that is passed as an argument

```
const higherOrderFunction = (callback)⟶ {
return callback()
}
```

Fig: Callback function

A callback is a JavaScript function passed to another function as an argument or a parameter. This function is to be executed whenever the function that it is passed to gets executed.

13. How do you debug a JavaScript code?

All modern web browsers, such as Chrome, Firefox, etc., have an inbuilt debugger that can be accessed anytime by pressing the relevant key, usually the F12 key. The debugging tools offer several features.

We can also debug JavaScript code inside a code editor to develop a JavaScript application, such as Visual Studio Code, Atom, Sublime Text, etc.

14. What is the difference between Function declaration and Function expression?

Function declaration	Function expression
Declared as a separate statement within the main JavaScript code	Created inside an expression or some other construct
Can be called before the function is defined	Created when the execution point reaches it; can be used only after that
Offers better code readability and better code organization	Used when there is a need for a conditional declaration of a function

Example:
function abc() {
return 5;
}
	Example:
	var a = function abc() {
	return 5;
	}

15. What are the ways of adding JavaScript code in an HTML file?

There are primarily two ways of embedding JavaScript code:

- We can write JavaScript code within the script tag in the same HTML file; this is suitable when we need just a few lines of scripting within a web page.
- We can import a JavaScript source file into an HTML document; this adds all scripting capabilities to a web page without cluttering the code.

Intermediate JavaScript Interview Questions and Answers

Here are some intermediate-level JavaScript interview questions and answers for you to prepare for your interviews.

16. What do you understand about cookies?

A cookie is generally a small piece of data sent from a website and stored on the user’s machine by a web browser that was used to access the website. Cookies are used to remember information for later use and record the browsing activity on a website.

17. How would you create a cookie?

The simplest way of creating a cookie using JavaScript is as follows:

document.cookie = "key1 = value1; key2 = value2; expires = date";

Did You Know? 🔍

Recruiters worldwide indicate that full-stack and back-end developers are among the most in-demand IT roles, with JavaScript being a key skill for these positions.

18. How would you read a cookie?

Reading a cookie using JavaScript is also very simple. We can use the document.cookie string that contains the cookies that we just created using that string.

The document.cookie string keeps a list of name-value pairs separated by semicolons, where ‘name’ is the cookie's name, and ‘value’ is its value. We can also use the split() method to break the cookie value into keys and values.

19. How would you delete a cookie?

We can just set an expiration date and time to delete a cookie. Specifying the correct path of the cookie we want to delete is a good practice since some browsers won’t allow the deletion cookies unless there is a clear path that tells which cookie to delete from the user’s machine.

```
function delete_cookie(name) {

  document.cookie = name + "=; Path=/; Expires=Thu, 01 Jan 1970 00:00:01 GMT;";

}
```

20. What’s the difference between let and var?

Both let and var are used for variable and method declarations in JavaScript. So there isn’t much of a difference between these two besides that while the var keyword is scoped by function, a block scopes the let keyword.

21. What are Closures in JavaScript?

Closures provide a better, concise way to write JavaScript code for the developers and programmers. Closures are created whenever a variable defined outside the current scope is accessed within the current scope.

```
function hello(name) {
  var message = "hello " + name;
  return function hello() {
    console.log(message);
  };
}
//generate closure
var helloWorld = hello("World");
//use closure
helloWorld();
```

22. What are the arrow functions in JavaScript?

Arrow functions are a short way of writing functions in JavaScript. The general syntax of an arrow function is as follows:

```
const helloWorld = () => {

  console.log("hello world!");

};
```

23. What are the different ways an HTML element can be accessed in a JavaScript code?

Here are the ways an HTML element can be accessed in a JavaScript code:

- getElementByClass(‘classname’): Gets all the HTML elements with the specified classname.
- getElementById(‘idname’): Gets an HTML element by its ID name.
- getElementbyTagName(‘tagname’): Gets all the HTML elements that have the specified tagname.
- querySelector(): Takes CSS style selector and returns the first selected HTML element.

24. What are the ways of defining a variable in JavaScript?

There are three ways of defining a variable in JavaScript:

- **Var**

This is used to declare a variable, and the value can be changed later within the JavaScript code.

- **Const**

We can also use this to declare/define a variable, but the value, as the name implies, is constant throughout the JavaScript program and cannot be modified later.

- **Let**

This mostly implies that the values can be changed later within the JavaScript code.

25. What are Imports and Exports in JavaScript?

Imports and exports help in writing modular code for our JavaScript applications. With the help of imports and exports, we can split JavaScript code into multiple files in a project. This greatly simplifies the application source code and encourages code readability.

```
calc.js
export const sqrt = Math.sqrt;
export function square(x) {
  return x * x;
}
export function diag(x, y) {
  return sqrt(square(x) + square(y));
}
```

This file exports two functions that calculate the squares and diagonal of the input, respectively.

```
main.js
import { square, diag } from "calc";
console.log(square(4)); // 16
console.log(diag(4, 3)); // 5
```

Therefore, here, we import those functions and pass input to those functions to calculate square and diagonal.

26. What is the difference between Document and Window in JavaScript?

Document	Window
The document comes under the Windows object and can be considered its property.	A window in JavaScript is a global object that holds the structure like variables, functions, location, history, etc.

27. What are some of the JavaScript frameworks and their uses?

JavaScript has a collection of many frameworks that aim to fulfill the different aspects of the web application development process. Some of the prominent frameworks are:

- React - Frontend development of a web application
- Angular - Frontend development of a web application
- Node - Backend or server-side development of a web application

28. What is the difference between Undefined and Undeclared in JavaScript?

Undefined	Undeclared
Undefined means a variable has been declared, but a value has not yet been assigned to that variable.	Variables that are not declared or that do not exist in a program or application.

29. What is the difference between Undefined and Null in JavaScript?

Undefined	Null

Undefined means a variable has been declared, but a value has not yet been assigned to that variable.	Null is an assignment value that we can assign to any variable meant to contain no value.

30. What is the difference between Session storage and Local storage?

Session storage	Local storage
The data stored in session storage expires or is deleted when a page session ends.	Websites store some data in local machines to reduce loading time; this data does not get deleted at the end of a browsing session.

31. What are the various data types that exist in JavaScript?

Javascript consists of two data types: primitive data types and non-primitive data types.

- Primitive Data types: These data types store a single value. Following are the sub-data t[illegible] in the Primitive data type.

- Boolean Data Types: It stores true and false values.

Example:

```
var a = 3;

var b =  4;

var c = 3;

(a == b) // returns false

(a == c) //returns true
```

- Null data Types: It stores either empty or unknown values.

Example:

```
var z = null;
```

- Undefined data Types: It stores variables that are only declared, but not defined or initialized.

Example:

```
var a; // a is undefined

var b = undefined; // we can also set the value of b variable as undefined
```

- Number Data Types: It stores integer as well as floating-point numbers.

Example:

```
var x = 4; //without decimal

var y = 5.6; //with decimal
```

- String data Types: It stores characters and alphanumeric values.

Example:

```
var str = "Raja Ram Mohan"; //using double quotes
```

var str2 = 'Raja Rani'; //using single quotes

- Symbols Data Types: It stores unique identifiers for objects.

Example:

var symbol1 = Symbol('symbol');

- BigInt Data Types: It stores the Number data types that are large integers and are above the limitations of number data types.

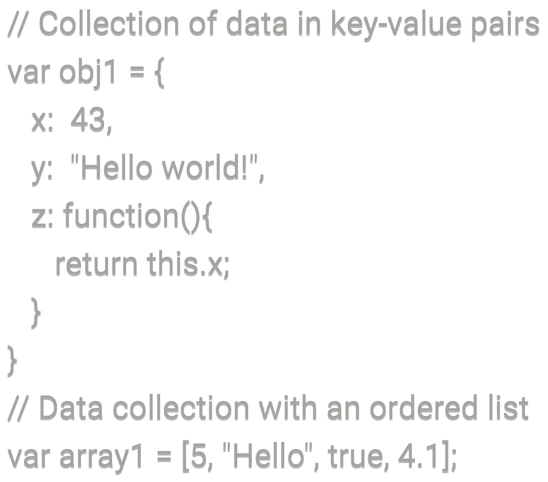
Example:

var bigInteger = 234567890123456789012345678901234567890;

- Non-Primitive Data Types

Non-Primitive data types are used to store multiple as well as complex values.

Example:



```
// Collection of data in key-value pairs
var obj1 = {
  x:  43,
  y:  "Hello world!",
  z: function(){
    return this.x;
  }
}
// Data collection with an ordered list
var array1 = [5, "Hello", true, 4.1];
```

32. What is the ‘this’ keyword in JavaScript?

The Keyword ‘this’ in JavaScript calls the current object as a constructor to assign values to object properties.

33. What is the difference between Call and Apply? (explain in detail with examples)

- Call

The call uses arguments separately.

Example:

```
function sayHello()
{
  return "Hello " + this.name;
}
var obj = {name: "Sandy"};
sayHello.call(obj);
// Returns "Hello Sandy"
```

- Apply

Apply uses an argument as an array.

Example:

```
function saySomething(message)
{
  return this.name + " is " + message;
}
var person4 = {name:  "John"};
saySomething.apply(person4, ["awesome"]);
```

34. What are the scopes of a variable in JavaScript?

The scope of variables in JavaScript determines the accessibility of variables and functions at various parts of one’s code. There are three types of scopes of a variable: global scope, function scope, block scope.

- Global Scope: It is used to access the variables and functions from anywhere inside the code.

Example:

```
var globalVariable = "Hello world";
function sendMessage(){
  return globalVariable; // globalVariable is accessible as it's written in global space
}
function sendMessage2(){
  return sendMessage(); // sendMessage function is accessible as it's written in global space
}
sendMessage2();  // Returns “Hello world”
```

- Function scope: It is used to declare the function and variables inside the function itself and not outside.

Example:

```
function awesomeFunction()
{
  var a = 3;
  var multiplyBy3 = function()
{
    console.log(a*3); // Can access variable "a" since a and multiplyBy3 both are written inside th
  }
}
console.log(a); // a is written in local scope and can't be accessed outside and throws a referen
multiplyBy3(); // MultiplyBy3 is written in local scope thus it throws a reference error
```

- Block Scope: It uses let and const to declare the variables.

Example:

```
{
  let x = 45;
}
console.log(x); // Gives reference error since x cannot be accessed outside of the block
for(let i=0; i<2; i++){
  // do something
}
console.log(i); // Gives reference error since i cannot be accessed outside of the for loop block
```

35. What are the arrow functions in JavaScript?

Arrow functions are used to write functions with a short and concise syntax. Also, it does not require a function keyword for declaration. An arrow function can be omitted with curly braces { } when we have one line of code.

Syntax of an arrow function:

```
const helloWorld = () => {
  console.log("hello world!");
};
Example:
// Traditional Function Expression
var add = function(a,b)
{
  return a + b;
}
// Arrow Function Expression
var arrowAdd = (a,b) => a + b;
```

36. Explain Hoisting in javascript. (with examples)

Hoisting in javascript is the default process behavior of moving the declaration of all the variables and functions on top of the scope where scope can be local or global.

Example 1:

```
hoistedFunction();  // " Hi There! " is an output that is declared as a function even after it is called

function hoistedFunction(){

  console.log(" Hi There! ");

}
```

Example 2:

```
hoistedVariable = 5;

console.log(hoistedVariable); // outputs 5 though the variable is declared after it is initialized

var hoistedVariable;
```

37. Difference between “ == “ and “ === “ operators (with examples)

1. “==” operator is a comparison operator that used to compare the values
2. “===” operator is also a comparison operator used to compare the values and types.

Example:

var x = 3;

var y = "3";

(x == y) // it returns true as the value of both x and y is the same

(x === y) // it returns false as the typeof x is "number" and typeof y is "string"

38. Difference between var and let keyword

- Keyword “var”
- In JavaScript programming, the “var” keyword has been used from the very initial stages of JavaScript.
- We can perform functions with the help of the keyword “var’ by accessing various variables.
- Keyword “let”
- The Keyword “let” was added later in ECMAScript 2015 in JavaScript Programming.
- Variable declaration is minimal with the help of the “let” keyword declared in Block. Also, it might result in a ReferenceError as the variable was declared in the “temporal dead zone” at the beginning of the block.

39. Implicit Type Coercion in javascript (in detail with examples)

When the value of one data type is automatically converted into another, it is called Implicit type coercion in javascript.

- String coercion

Example:

```
var x = 4;

var y = "4";

x + y // Returns "44"
```

- Boolean coercion

Example:

```
var a = 0;

var b = 32;

if(a) { console.log(a) }   // This code will run inside the block as the value of x is 0(Falsy)

if(b) { console.log(b) }    // This code will run inside the block as the value of y is 32 (Truthy)
```

40. Is javascript a statically typed or a dynamically typed language?

Yes, JavaScript is a dynamically typed language and not statically.

41. NaN property in JavaScript

NaN property in JavaScript is the “Not-a-Number” value that is not a legal number.

42. Passed by value and passed by reference

- Passed By Values Are Primitive Data Types.

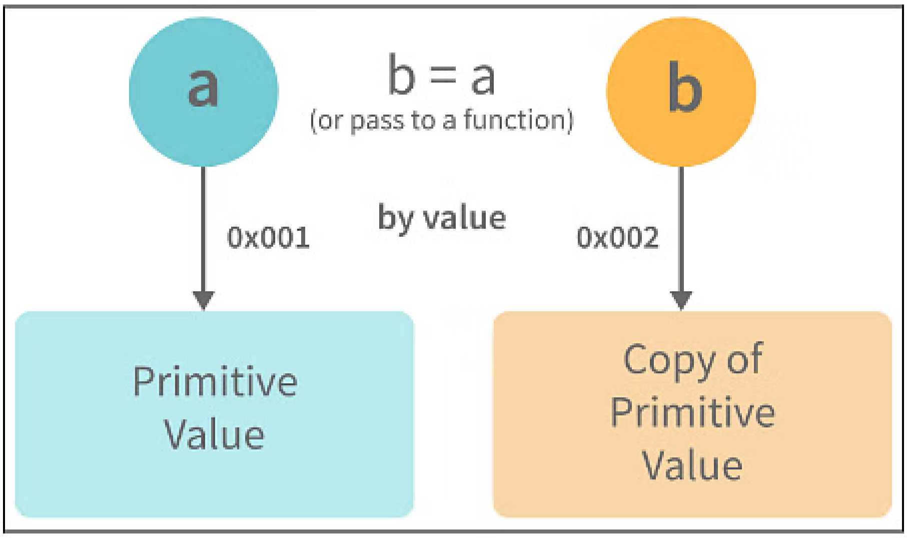
Consider the following example:

Here, the a=432 is a primitive data type i.e. a number type with an assigned value by the operator. When the var b=a code gets executed, the value of ‘var a’ returns a new address for ‘var b’ by allocating a new space in the memory so that ‘var b’ will be operated at a new location.

Example:

var a = 432;

var b = a;



- Passed by References Are Non-primitive Data Types.

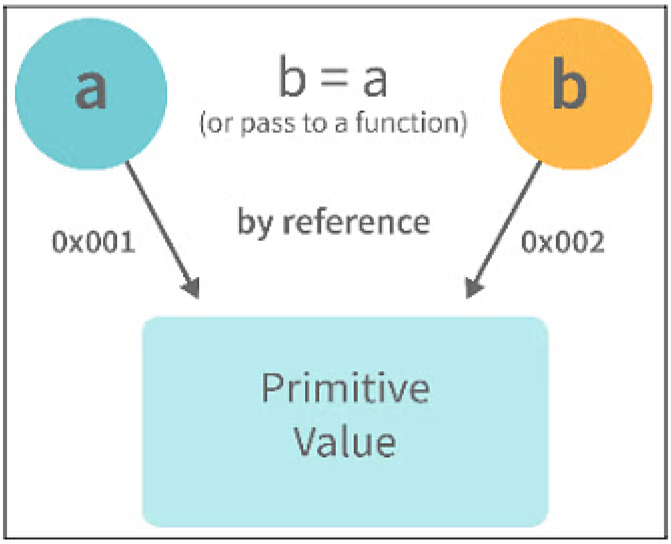
Consider the following example:

The reference of the 1st variable object i.e. ‘var obj’ is passed through the location of anoth variable i.e. ‘var obj2’ with the help of an assigned operator.

Example:

```
var obj = { name: "Raj", surname: "Sharma" };

var obj2 = obj;
```



43. Immediately Invoked Function in JavaScript

An Immediately Invoked Function, also abbreviated as IIFE or IIFY, runs as soon as it is defined. To run the function, it needs to be invoked; otherwise, the declaration of the function is returned.

Syntax:

```
(function()

{

 // Do something;

})

();
```

44. Characteristics of javascript strict-mode

- Strict mode does not allow duplicate arguments and global variables.
- In strict mode, one cannot use JavaScript keywords as a parameter or function name.

- In strict mode, one cannot use JavaScript keywords as a parameter or function name.
- All browsers support strict mode.
- Strict mode can be defined at the start of the script with the help of the keyword ‘use strict’.

45. Higher Order Functions (with examples)

Higher-order functions are the functions that take functions as arguments and return them by operating on other functions

Example:

```
function higherOrder(fn)

 {

  fn();

}

higherOrder(function() { console.log("Hello world") });
```

46. Self Invoking Functions

Self Invoking Functions is an automatically invoked function expression followed by (), where it does not need to be requested. Nevertheless, the declaration of the function cannot be invoked by itself.

47. difference between exec () and test () methods

- exec()
- It is an expression method in JavaScript used to search a string with a specific pattern.
- Once found, the pattern will be returned directly; otherwise, it will return an “empty” result.
- test ()
- It is an expression method in JavaScript used to search a string with a specific pattern or text.
- Once it has been found, the pattern will return the Boolean value 'true', else it returns ‘false’.

48. currying in JavaScript (with examples)

In JavaScript, when a function of an argument is transformed into functions of one or more arguments, it is called Currying.

Example:

```
function add (a) {

  return function(b){

    return a + b;

  }

}

add(3)(4)
```

49. Advantages of using External JavaScript

- External Javascript allows web designers and developers to collaborate on HTML and JavaScript files.
- It also enables you to reuse the code.
- External javascript makes Code readability simple.

50. What are object prototypes?

Following are the different object prototypes in JavaScript that are used to inherit particular properties and methods from the Object.prototype.

1. Date objects are used to inherit properties from the Date prototype
2. Math objects are used to inherit properties from the Math prototype
3. Array objects are used to inherit properties from the Array prototype.

51. Types of errors in javascript

Javascript has two types of errors: Syntax errors and Logical errors.

52. What is memoization?

In JavaScript, memoization is called memoization when we want to cache the return value of a function concerning its parameters. It is used to speed up the application, especially in case of complex, time-consuming functions.

53. Recursion in a programming language

Recursion is a programming language technique used to iterate over an operation, whereas a function calls itself repeatedly until we get the result.

54. Use of a constructor function (with examples)

Constructor functions are used to create single objects or multiple objects with similar properties and methods.

Example:

```
function Person(name,age,gender)
{
  this.name = name;
  this.age = age;
  this.gender = gender;
}
var person1 = new Person("Vivek", 76, "male");
console.log(person1);
var person2 = new Person("Courtney", 34, "female");
console.log(person2);
```

55. Which method retrieves a character from a certain index?

We can retrieve a character from a certain index with the help of charAt() function method.

56. What is BOM?

BOM is the Browser Object Model where users can interact with browsers that is a window, an initial object of the browser. The window object comprises a document, history, screen, navigator, location, and other attributes. Nevertheless, the window's function can be called directly and by referencing the window.

57. Difference between client-side and server-side

Client-side JavaScript

- Client-side JavaScript comprises fundamental language and predefined objects that perform JavaScript in a browser.
- Also, it is automatically included in the HTML pages where the browser understands the script.

Server-side Javascript

- Server-side JavaScript is quite similar to client-side JavaScript.
- Server-side JavaScript can be executed on a server.
- The server-side JavaScript is deployed once the server processing is done.

- The server-side JavaScript is deployed once the server processing is done.

58. What is the prototype design pattern?

The Prototype design Pattern is also known as a property or prototype pattern used to produce different objects and prototypes that are replicated from a template with a specific value.

59. Differences between declaring variables using var, let and const.

var	let	const
There is a global scope as well as a function scope.	There is neither a global scope nor a function scope.	There is neither a global sco nor a function scope.
There is no block scope.	There is no block scope.	There is no block scope.
It can be reassigned.	cIt cannot be reassigned.	It cannot be reassigned.

Example 1: Using ‘var’ and ‘let’ variable

```
var variable1 = 31;
let variable2 = 89;
let variable2 = 39;
function catchValues()
{
  console.log(variable1);
  console.log(variable2);
// Both the variables are accessible from anywhere as their declaration is in the global scope
}
window.variable1; // Returns the value 31
window.variable2; // Returns undefined
Example 2: Using ‘const’ variable
const x = {name:"Vijay"};
x = {address: "Mumbai"}; // Throws an error
x.name = "Radha"; // No error is thrown
const y = 31;
y = 44; // Throws an error
```

60. Rest parameter and spread operator

Rest Parameter(...)

- The rest parameter is used to declare the function with improved handling of parameters.
- Rest parameter syntax can be used to create functions to perform functions on the variable number of arguments.
- It also helps to convert any number of arguments into an array and helps in extracting some or all parts of the arguments.

Spread Operator(...)

- In a function call, we use the spread operator.
- It's also to spread one or more expected arguments in a function call.

- It's also to spread one or more expected arguments in a function call.
- The spread operator takes an array or an object and spreads them.

61. Promises in JavaScript

Promises in JavaScript have four different states. They are as follows:

Pending	Fulfilled	Rejected
Pending is an initial state of promise. It is the initial state of promise, which is in the pending state, that is neither fulfilled nor rejected.	The state where the promise has been fulfilled assures that the async operation is done.	It is the state where the pron is rejected, and the async operation has failed.

Example:

```
function sumOfThreeElements(...elements)
{
 return new Promise((resolve,reject)=>{
  if(elements.length > 3 )
{
   reject("Just 3 elements or less are allowed");
  }
  else
{
   let sum = 0;
   let i = 0;
   while(i < elements.length)
{
    sum += elements[i];
    i++;
   }
   resolve("Sum has been calculated: "+sum);
  }
 })
}
```

62. Classes in JavaScript

classes are syntactic sugars for constructor functions mentioned in the ES6 version of JavaScript. Classes are not hoisted-like Functions and can’t be used before it is declared. Also, it can inherit properties and methods from other classes with the help of extended keywords. An error will be shown if the strict mode (‘use strict’) is not followed.

63. What are generator functions?

Generator functions are declared with a special class of functions and keywords using function*. It does not execute the code. However, it returns a generator object and handles the execution.

64. What is WeakSet?

WeakSet is a collection of unique and ordered elements that contain only objects which are referenced weakly.

65. What is the use of callbacks?

- A callback function sends input into another function and is performed inside another function.
- It also ensures that a particular code does not run until another code has completed its execution.

66. What is a WeakMap?

Weakmap is referred to as an object having keys and values; if the object is without reference, it is collected as garbage.

67. What is Object Destructuring? (with examples)

Object destructuring is a method to extract elements from an array or an object.

Example 1: Array Destructuring

```
const arr = [1, 2, 3];
const first = arr[0];
const second = arr[1];
const third = arr[2];
```

Example 2: Object Destructuring

```
const arr = [1, 2, 3];
const [first,second,third,fourth] = arr;
console.log(first); // Outputs 1
console.log(second); // Outputs 2
console.log(third); // Outputs 3
```

68. Prototypal vs Classical Inheritance

Prototypal Inheritance

- Prototypal inheritance allows any object to be cloned via an object linking method. It serves as a template for those other objects, whether they extend the parent object or not.

Classical Inheritance

- Classical inheritance is a class that inherits from the other remaining classes.

69. What is a Temporal Dead Zone?

Temporal Dead Zone is a behavior that occurs with variables declared using let and const keywords before they are initialized.

70. JavaScript Design Patterns

When we build JavaScript browser applications, there might be chances to occur errors where JavaScript approaches it repetitively. This repetitive approach pattern is called the JavaScript design pattern. JavaScript design patterns consist of Creational Design Pattern, Structural Design Pattern, and Behavioral Design patterns.

71. Difference between Async/Await and Generators

Async/Await

- Async-await functions are executed sequentially, one after another, in an easier way.
- Async/Await function might throw an error when the value is returned.

Generators

- Generator functions are executed with one output at a time by the generator's yield by yield.
- The 'value: X, done: Boolean' is the output result of the Generator function.

72. Primitive data types

The primitive data types are capable of displaying one value at a time. It consists of Boolean, Undefined, Null, Number, and String data types.

73. Role of deferred scripts

The Deferred scripts are used for the HTML parser to finish before executing it.

74. What is Lexical Scoping?

Lexical Scoping in JavaScript can be performed when the internal state of the JavaScript

Lexical Scoping in JavaScript can be performed when the internal state of the JavaScript function object consists of the function’s code and references concerning the current scope chain.

75. What is this [[[]]]?

This ‘[[[]]]’ is a three-dimensional array.

76. Are Java and JavaScript the same?

Java and JavaScript are not identical; they are distinct programming languages with different purposes and characteristics. Java is a high-level, object-oriented programming language designed for building platform-independent applications, often used in enterprise environments for server-side applications, mobile applications, and large systems. It requires compilation and runs on the Java Virtual Machine (JVM). Conversely, JavaScript is a lightweight, interpreted scripting language primarily used to create dynamic and interactive website content. It runs directly in web browsers and is an essential technology for web development alongside HTML and CSS. Despite their similar names, their syntax, use cases, and execution environments are quite different.

77. How to detect the OS of the client machine using JavaScript?

The OS on the client machine can be detected with the help of navigator.appVersion string.

78. Requirement of debugging in JavaScript

- We can debug the code using web browsers such as Google Chrome and Mozilla Firefox.
- We can debug in JavaScript using two methods: console.log() and the debugger keywo

79. What are the pop-up boxes available in JavaScript?

Pop-up boxes available in JavaScript are Alert Box, Confirm Box, and Prompt Box.

Advanced JS Interview Questions and Answers

Here are some advanced-level JavaScript interview questions and answers for you to prepare for your interviews.

80. How do you empty an array in JavaScript?

There are a few ways in which we can empty an array in JavaScript:

- By assigning array length to 0:

```
var arr = [1, 2, 3, 4];

arr.length = 0;
```

- By assigning an empty array:

```
var arr = [1, 2, 3, 4];

arr = [];
```

- By popping the elements of the array:

```
var arr = [1, 2, 3, 4];

while (arr.length > 0) {

  arr.pop();

}
```

- By using the splice array function:

```
var arr = [1, 2, 3, 4];

arr.splice(0, arr.length);
```

81. What is the difference between Event Capturing and Event Bubbling?

Event Capturing	Event Bubbling
This process starts with capturing the event of the outermost element and then propagating it to the innermost element.	This process starts with capturing the event of the innermost element and then propagating it to the outermost element.

82. What is the Strict mode in JavaScript?

Strict mode in JavaScript introduces more stringent error-checking in JavaScript code.

- While in Strict mode, all variables have to be declared explicitly; values cannot be assigned to a read-only property, etc.
- We can enable strict mode by adding ‘use strict’ at the beginning of a JavaScript code c within a specific code segment.

83. What would be the output of the below JavaScript code?

var a = 10;

if (function abc(){})

{

a += typeof abc;

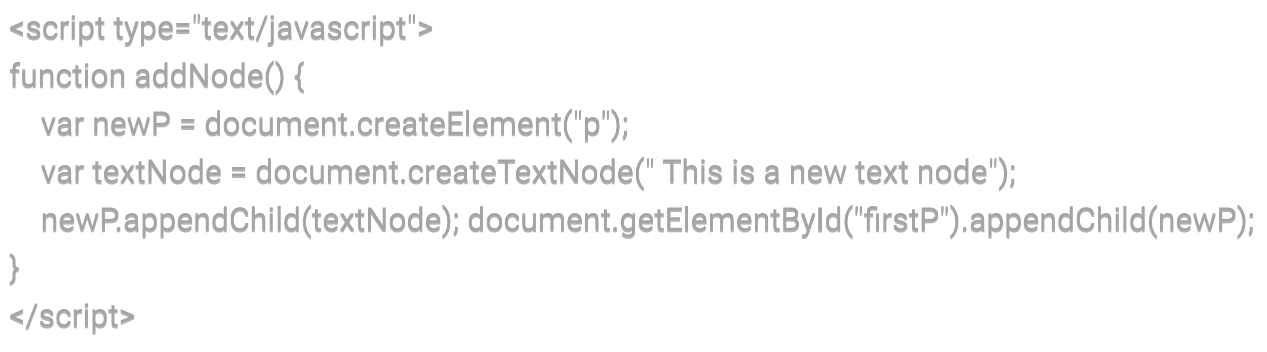
}

console.log(a);

The output of this JavaScript code will be 10undefined. The if condition statement in the code evaluates using eval. Hence, eval(function abc(){}) will return function abc(){}.

Inside the if statement, executing typeof ABC returns undefined because the if statement code executes at run time while the statement inside the if condition is being evaluated.

84. Can you write a JavaScript code for dynamically adding new elements?



```
<script type="text/javascript">
function addNode() {
   var newP = document.createElement("p");
   var textNode = document.createTextNode(" This is a new text node");
   newP.appendChild(textNode); document.getElementById("firstP").appendChild(newP);
}
</script>
```

85. What is the difference between Call and Apply?

Call	Apply
In the call() method, arguments are provided individually and a ‘this’ value.	In the apply() method, arguments are provided in an array along with a ‘this’ value.

86. What will be the output of the following code?

```
var Bar = Function Foo()

{

return 11;

};

typeof Foo();
```

The output would be a reference error since a function definition can only have a single reference variable as its name.

87. What will be the output of the following code?

```
var Student = {

  college: "abc",

};

var stud1 = Object.create(Student);

delete stud1.college;
```

console.log(stud1.company);

This is essentially a simple example of object-oriented programming. Therefore, the output will be ‘abc’ as we are accessing the property of the student object.

88. How do you remove duplicates from a JavaScript array?

There are two ways in which we can remove duplicates from a JavaScript array:

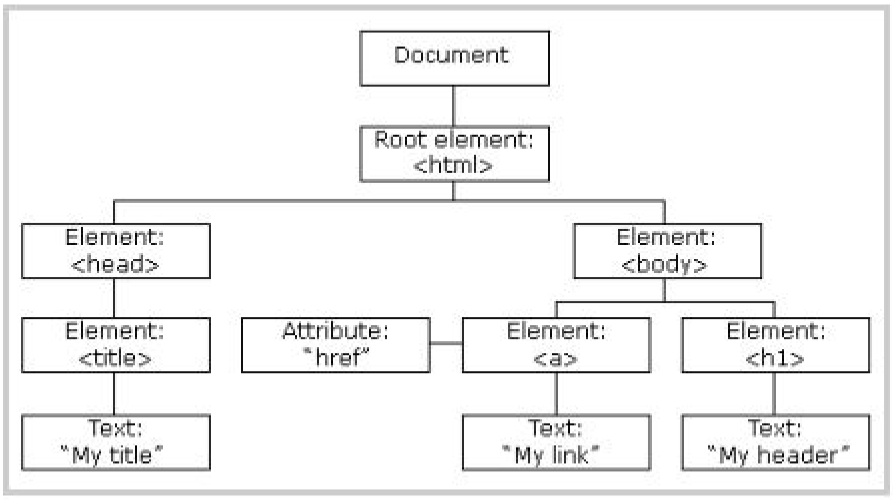
By using the Filter Method

To call the filter() method, three arguments are required. These are, namely, array, current element, and index of the current element.

By using the For Loop

An empty array is used to store all the repeating elements.

89. Can you draw a simple JavaScript DOM (Document Object Model)?



As you prepare for your upcoming job interview, we hope these JavaScript Interview Questions and answers have provided more insight into what types of questions you will likely be asked.

90. State the use case of anonymous functions.

The anonymous functions are unnamed and used as arguments for functional programming. They also find usage for code encapsulation to avoid variable leakage in the global scope, f immediate execution, as callbacks, as arguments in high-order functions or Lodas and eve handling.

Handling events like clicks and keyboard inputs is a fundamental JavaScript skill. In React, event handling is slightly different due to the virtual DOM. If you're curious about implementing event-driven components, check out our step-by-step tutorial on how to create a functional React dropdown menu.

91. What is functional programming?

Functional programming is the concept that refers to program development via mathematical functions and expressions. Here, the codes do not involve data or state mutation. Functional programming is preferred owing to the generation of bug-fee code and easy readability and maintainability.

92. Differentiate between Object.freeze() and const

Object.freeze() removes the possibility of modifications in the object. The object can not be added or removed. Using the function includes taking the object as an argument and returning the immutable object. On the other hand, const is similar to let, and it binds variables. The variable declaration performed by const makes them block-scoped.

93. What is an event loop in JavaScript?

It refers to a JavaScript process that allows multiple task handling without impact on the main thread. The event loop performs code execution, event collection and processing and queued sub-task execution. Its ability to execute non-blocking and asynchronous operations ensures smooth running of the code.

94. What is bind() in JavaScript?

The bind function in JavaScript allows method borrowing from one object to another. It also preserves ‘this’. The syntax of the function is function.bind(thisArg, arg1, arg2, ...).

95. Differentiate between .foreach() and .map().

While both methods lead to iterations in arrays, there is a difference between them. The .foreach() performs array iteration by executing each element-based specified function only once. It is used when dealing with operations like element logging and modification. The syntax of .foreach() is array.forEach(callback(currentValue, index, array), thisArg);.

The .map() generates a new array by applying a specific function to each element in the original

The .map() generates a new array by applying a specific function to each element in the original array. Here, the users can expect a new array with modified values in output without impacting the original array. The common usage is for data transformation and manipulation. The syntax of the array is map((currentElement, indexOfElement, array) => { ... });.

96. Differentiate between shallow and deep copy in JavaScript.

The shallow copy refers to copying the object reference to a new variable. The nested arrays or objects refer to the original memory location, but there are changes in top-level properties. It works by the usage of the assignment operator (=). The deep copy refers to the independent copying of the object. Here, the changes do not affect the different objects. Developing deep copy includes using JSON.parse() and JSON.stringify() methods.

97. State the purpose of Array.prototype.map() and Array.prototype.filter().

Array's map() method generates a new array comprising the result of calling a provided function on every element in the calling array. The syntax used here is

map(callbackFn)

map(callbackFn, thisArg)

Array's filter() method generates a shallow copy of the specific part of the given array. The created copy comprises the elements that pass the test implemented by the provided function. The syntax used here is

filter(callbackFn)

filter(callbackFn, thisArg)

98. What is the setInterval() method?

Previous Next

Tutorial Playlist

innerHTML gets plain text or HTML content, including HTML tags. The innerText is used to receive plain text in return. The text is of a particular node, and descendants are stated in the input.

JavaScript Coding Challenges for Interviews

JavaScript interview questions can also include coding challenges. Here is how to complete the same:

1. Write a program to reverse a given integer number.

The program to reverse a given integer will be:

```
def reverse_integer(num):
    is_negative = num < 0
    num = abs(num)
    reversed_num = 0
    while num != 0:
        digit = num % 10
        reversed_num = reversed_num * 10 + digit
        num = num // 10
    if is_negative:
```

```
    if is_negative:
        reversed_num = -reversed_num
    return reversed_num
number = 12345
reversed_number = reverse_integer(number)
print(f"The reversed number of {number} is {reversed_number}")
```

2. Write a program to swap two integers without using a temporary variable. Name the number variables as num1 and num2.

The program to swap two integers with the stated condition is:

```
num1 = 5
num2 = 10
num1 = num1 + num2
num2 = num1 - num2
num1 = num1 - num2
print("num1:", num1)
print("num2:", num2)
```

3. Write a program to perform a deepequal check.

The program to perform a deepequal check is:

```
function deepEqual(obj1, obj2) {
    if (obj1 === obj2) return true;
    if (typeof obj1 !== 'object' || typeof obj2 !== 'object') return false;
    const keys1 = Object.keys(obj1);
    const keys2 = Object.keys(obj2);
    if (keys1.length !== keys2.length) return false;
    for (let key of keys1) {
        if (!keys2.includes(key)) return false;
        if (!deepEqual(obj1[key], obj2[key])) return false;
    }
    return true;
}
```

4. Write a program to calculate the factorial of a given number (n)?

The program for factorial calculation is:

```
function factorial(n) {
   if (n === 0 || n === 1) return 1;
   return n * factorial(n - 1);
}
let number = 5;
console.log(factorial(number));
```

5. Write a program to find the smallest number in a nested array.

The program for finding the smallest number in a nested array is:

```
function findSmallest(arr) {
   let flatArray = arr.flat(Infinity);
   return Math.min(...flatArray);
}
let nestedArray = [ [3, 5, 7], [2, 8, [1, 4]], [6, 9] ];
console.log(findSmallest(nestedArray));
```

How to Prepare for a JavaScript Interview?

Preparation for JavaScript interview questions and answers must be a strategic process. Here are the steps to focus on:

- Begin with learning about the fundamentals. You must learn and improve your understanding of fundamentals such as data types, variables, arrays, functions, operators, Object-Oriented Programming (OOP) and others.
- Practice coding to determine your level of application of the known concepts.
- Go through the possible technical interview questions.
- Remember to enhance your communication skills and prepare answers to behavioral and other questions.
- Learn about the company, the position you are applying for and job and company-specifi problems.

What Are Some Mistakes to Avoid in JavaScript Interviews?

Some mistakes that commonly occur while dealing with JavaScript are:

- Confusion between the assignment and comparison operator.

The assignment operator is marked by (=), a single equal sign. Conversely, the comparison operation is indicated by (==), which is a double equal sign.

- Confusion between a function expression and declaration.

Function expression means the function can not be called before assigning, while function declaration is the opposite of it. The function declaration allows calling the function before assigning.

- Ignoring asynchronous patterns.

The asynchronous patterns result in delays, lack of user interface response and event loop block. To avoid such an error, use async/await, which allows proper handling.

- Ignoring possible errors.

It leads to wrong results and even crashes. Using try/catch blocks allows better error handling.

- Wrong usage of ‘this’.

‘This’ in JavaScript is based on the called function. It does not imply the present object, so understanding the specifications is necessary. To avoid the issue, merge ‘this’ with apply(), call() or bind(). Alternatively, users can use arrow functions without their own ‘this’.

Get Ahead of the Curve and Master JavaScript Today

Are you wondering how to gain the skills necessary to take advantage of JavaScript's immense popularity now that you are familiar with JS Interview Questions and Answers? We have got your back! We offer comprehensive Java Certification Training, which will help you get a job as a software engineer upon completion.

To learn more, check out our YouTube video that briefly introduces JavaScript Interview Questions and answers and helps clear doubts for your next JavaScript interview. If you're an aspiring web and mobile developer, JavaScript training will broaden your skills and career horizons.

Want to ace the upcoming JavaScript interview? There's nowhere else to look! We guarantee you're ready to wow prospective employers with our extensive list of the Top 90+ JavaScript Interview Questions and Answers, which covers everything from fundamental ideas to complex subjects. Our Full Stack Java Developer will help you brush up on your JavaScript knowledge and confidently approach any interview, regardless of experience level.

Bonus Read: Coding Interview Q&As you must prepare for a successful interview in 2025.

FAQs

1. Should I focus more on ES6+ features for interviews?

The promising impact of ES6+, which increases readability, maintainability, and productivity, makes it essential for any eligible candidate to be well aware of. Some critical features that make it worthy are arrow functions, symbols, promises, and Let and const block scoping.

2. How do I approach a coding challenge during a JavaScript interview?

The perfect approach is learning and application through practice. Dealing with the questic helps your mind adapt to the concept and program and develop answers accordingly. Develop the codes for varying problems to qualify for the interview.

3. How important is it to know about this and closures?

‘This’ is a fundamental concept in JavaScript that refers to a current object in the executed function. It is often associated with confusion among beginners, so concept clarity is essential. Closures are another vital aspect of understanding variables and functions and functional programming.

4. Do I need to know about TypeScript for JavaScript interviews?

The requirement to know about TypeScript in JavaScript interviews depends on job requirements. Also, knowledge of JavaScript eases learning TypeScript, so learning it is recommended.

5. What is the difference between synchronous and asynchronous JavaScript, and why is it important?

Synchronous Javascript refers to code execution in sequential order, while asynchronous means simultaneously running multiple tasks for execution. Both hold importance owing to specific characteristics like simplicity and predictability of synchronous and effective responsiveness and performance of asynchronous JavaScript.

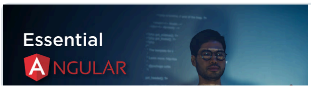
About the Author

Haroon Ahamed Kitthu

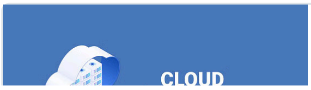
Haroon is the Senior Associate Director of Products at Simplilearn. bringing 10 years of expertise in product management and software development. He excels in building customer-f…

View More

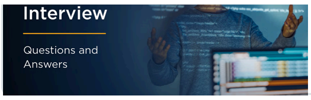
Recommended Resources



45 Angular Interview Questions in 2025: Pre



Cloud Computing
Guide



Questions in 2025: Pre...

Guide