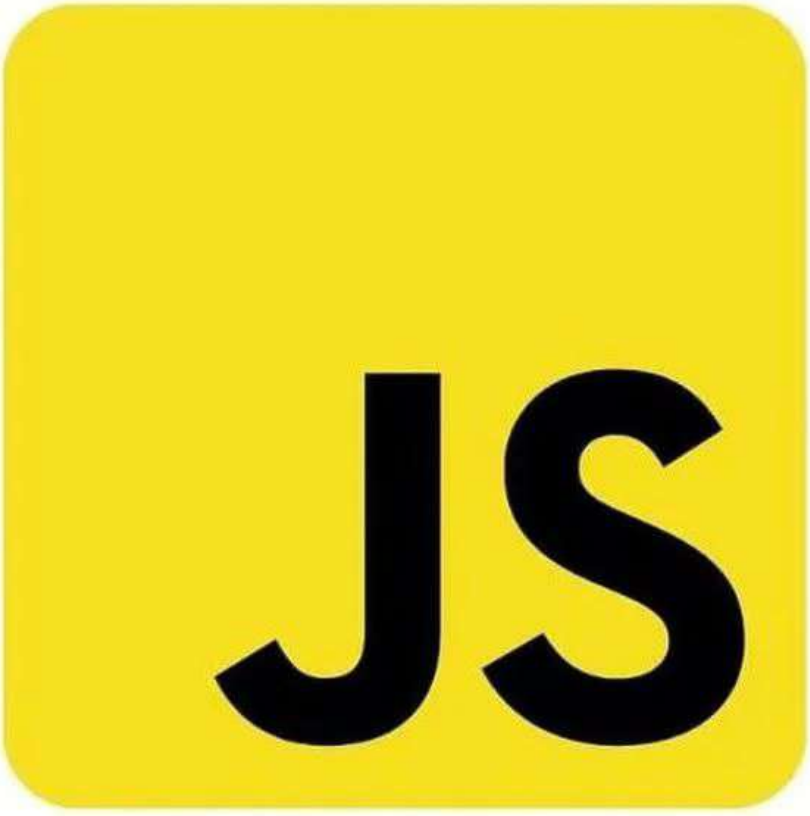
50 JavaScript Interview Q/A



1. What is JavaScript?

JavaScript is a high-level, interpreted programming language primarily used for creating interactive web pages.

2. What are the key features of JavaScript?

Dynamic typing, prototype-based object-orientation, first-class functions, and lexical scoping are some key features of JavaScript.

3. What are the data types in JavaScript?

JavaScript has six primitive data types: **string**, **number**, **boolean**, **null**, **undefined**, and **symbol** (added in ECMAScript 6). Objects are also a data type, including arrays and functions.

4. What is the difference between null and undefined?

null represents the intentional absence of any object value, while **undefined** represents the absence of a value or the value has not been assigned yet.

5. What is the difference between == and === operators?

== checks for equality of values after type coercion, while **===** checks for equality of values without type coercion, also known as strict equality.

6. What is closure in JavaScript?

A closure is a function defined inside another function (the outer function) and has access to the outer function's variables. It has access to its own scope, the outer function's scope, and the global scope.

7. Explain event delegation in JavaScript.

Event delegation is a technique where you attach an event listener to a parent element instead of multiple child elements. When an event occurs, it bubbles up to the parent element where you can handle it based on the target.

8. What is a callback function?

A callback function is a function passed as an argument to another function to be executed later. It is commonly used in asynchronous programming, event handling, and functional programming.

9. What is the difference between null and undefined?

null represents the intentional absence of any object value, while **undefined** represents the absence of a value or the value has not been assigned yet.

10. Explain the concept of hoisting.

Hoisting is a JavaScript mechanism where variables and function declarations are moved to the top of their containing scope during the compilation phase, before the code execution.

11. What is the difference between let, const, and var?

var is function-scoped, **let** and **const** are block-scoped. **var** variables can be redeclared and updated, while **let** variables can be updated but not redeclared, and **const** variables cannot be updated or redeclared.

12. What is event bubbling and capturing in JavaScript?

Event bubbling is the process where the event starts at the innermost target element and then bubbles up the DOM hierarchy until it reaches the root element. Event capturing is the opposite, where the event starts at the root element and then trickles down to the target element.

13. What is the difference between null and undefined?

null represents the intentional absence of any object value, while **undefined** represents the absence of a value or the value has not been assigned yet.

14. Explain the this keyword in JavaScript.

this refers to the object to which a function or method belongs, depending on how the function is called. It can change its value based on how a function is invoked.

15. What is event delegation in JavaScript?

Event delegation is a technique where you attach an event listener to a parent element instead of multiple child elements. When an event occurs, it bubbles up to the parent element where you can handle it based on the target.

16. What is a closure in JavaScript?

A closure is a function defined inside another function (the outer function) and has access to the outer function's variables. It has access to its own scope, the outer function's scope, and the global scope.

17. What is the difference between null and undefined?

null represents the intentional absence of any object value, while **undefined** represents the absence of a value or the value has not been assigned yet.

18. What is the difference between == and === operators?

== checks for equality of values after type coercion, while **===** checks for equality of values without type coercion, also known as strict equality.

19. What is the event loop in JavaScript?

The event loop is a mechanism in JavaScript that handles asynchronous operations. It continuously checks the call stack and the task queue, moving functions from the queue to the stack when the stack is empty.

20. Explain the difference between function declaration and function expression.

Function declarations are hoisted, meaning they are available before the code executes. Function expressions are not hoisted and are available only after the interpreter reaches the line of code where they are defined.

21. What is the difference between null and undefined?

null represents the intentional absence of any object value, while **undefined** represents the absence of a value or the value has not been assigned yet.

22. What is prototypal inheritance in JavaScript?

Prototypal inheritance is a mechanism where an object inherits properties and methods from its prototype object. Each object has a prototype object, and properties/methods not found in the object itself are searched for in its prototype chain.

23. What is the difference between apply, call, and bind methods?

call and **apply** are used to invoke a function with a specified **this** value and arguments. The only difference is in how arguments are passed: **call** accepts an argument list, while **apply** accepts an array of arguments. **bind** returns a new function with the specified **this** value and arguments bound to it, but it doesn't immediately invoke the function.

24. What is a promise in JavaScript?

A promise is an object representing the eventual completion or failure of an asynchronous operation. It allows you to handle asynchronous operations more elegantly than using callbacks and provides methods for handling success and failure.

25. What is a generator function in JavaScript?

A generator function is a special type of function that can be paused and resumed. It's defined using the **function*** syntax and uses the **yield** keyword to yield values one at a time, allowing for more control over asynchronous operations.

26. What is the purpose of the use strict directive in JavaScript?

The use strict directive enables strict mode, which imposes stricter parsing and error handling on your code. It helps catch common coding mistakes and makes JavaScript behave more predictably.

27. What is the Map object in JavaScript?

The Map object is a collection of key-value pairs where keys can be of any type. It allows you to store and retrieve data based on keys, and preserves the insertion order of the elements.

28. What is the Set object in JavaScript?

The Set object is a collection of unique values where each value can occur only once. It provides methods for adding, removing, and checking for the presence of elements.

29. What is the Symbol data type in JavaScript?

Symbol is a primitive data type introduced in ECMAScript 6. It represents a unique identifier and can be used as an object property key.

30. What is the async/await feature in JavaScript?

async/await is a syntactic sugar for writing asynchronous code in a synchronous-like manner. async functions return a promise, and await can be used inside async functions to pause execution until a promise is resolved.

31. What are arrow functions in JavaScript?

Arrow functions are a concise way to write anonymous functions in JavaScript. They have a shorter syntax compared to traditional function expressions and do not bind their own this, arguments, super, or new.target.

32. Explain the difference between let and const in JavaScript.

Both let and const are block-scoped declarations introduced in ECMAScript 6. The main difference is that variables declared with let can be reassigned, while variables declared with const cannot be reassigned; however, the value they hold can still be mutable for non-primitive types.

33. What are template literals in JavaScript?

Template literals are a way to create strings in JavaScript using backticks (`) instead of single or double quotes. They support interpolation, allowing you to embed expressions and variables directly within the string.

34. What is a destructuring assignment in JavaScript?

Destructuring assignment is a syntax that allows you to extract values from arrays or objects and assign them to variables in a concise way. It can also be used with function parameters to extract values from objects passed as arguments.

35. What is the spread operator in JavaScript?

The spread operator (...) allows an iterable (like an array or string) to be expanded into individual elements. It is commonly used for array literals, function arguments, and object literals.

36. What is the Object constructor in JavaScript?

The **Object** constructor creates an object wrapper for the given value. If the value is **null** or **undefined**, it will create and return an empty object. Otherwise, it will return an object of a type that corresponds to the given value.

37. What are the different ways to create objects in JavaScript?

Objects can be created using object literals ({}), the **Object** constructor (**new Object()**), or constructor functions with the **new** keyword. In ECMAScript 6, you can also use classes to create objects.

38. What is a higher-order function in JavaScript?

A higher-order function is a function that takes one or more functions as arguments or returns a function as its result. It enables functions to be used as values, allowing for more flexible and expressive code.

39. Explain the concept of currying in JavaScript.

Currying is a technique in functional programming where a function with multiple arguments is transformed into a sequence of nested functions, each taking a single argument. It allows you to partially apply a function and create new functions from it.

40. What is memoization in JavaScript?

Memoization is an optimization technique used to cache the results of expensive function calls and return the cached result when the same inputs occur again. It can improve the performance of functions by avoiding unnecessary computations.

41. What is the typeof operator in JavaScript?

The **typeof** operator is used to determine the data type of a variable or expression. It returns a string indicating the data type, such as **'string'**, **'number'**, **'boolean'**, **'object'**, **'function'**, or **'undefined'**.

42. What is the purpose of the Array map() method in JavaScript?

The **map()** method creates a new array by applying a function to each element of the original array. It does not mutate the original array and returns a new array with the results of the function applied to each element.

43. What is the purpose of the Array filter() method in JavaScript?

The **filter()** method creates a new array with all elements that pass the test implemented by the provided function. It does not mutate the original array and returns a new array containing only the elements that satisfy the condition.

44. What is the purpose of the Array reduce() method in JavaScript?

The **reduce()** method applies a function to each element of the array, resulting in a single value. It iterates over the array from left to right and accumulates a single value by applying the provided function to each element.

45. What is the purpose of the Array forEach() method in JavaScript?

The **forEach()** method executes a provided function once for each array element. It does not return a new array but is useful for performing side effects or iterating over array elements.

46. What is the difference between for...of and for...in loops in JavaScript?

The **for...of** loop iterates over iterable objects (like arrays, strings, maps, sets, etc.) and returns the values of each iteration. The **for...in** loop iterates over the properties of an object and returns the property names.

47. What is the purpose of the Array find() method in JavaScript?

The **find()** method returns the first element in the array that satisfies the provided testing function. It returns **undefined** if no matching element is found.

48. What is the purpose of the Array some() method in JavaScript?

The **some()** method tests whether at least one element in the array passe the provided testing function. It returns **true** if at least one element satisfi the condition, otherwise **false**.

49. What is the purpose of the Array every() method in JavaScript?

The **every()** method tests whether all elements in the array pass the provided testing function. It returns **true** if all elements satisfy the condition, otherwise **false**.

50. What is the purpose of the Array sort() method in JavaScript?

The **sort()** method sorts the elements of an array in place and returns the sorted array. By default, it sorts elements as strings and converts element to strings before comparing them. Optionally, you can provide a compare function to define the sorting order.