1. **What is JavaScript?**

JavaScript is a programming language that is primarily used for creating interactive web pages and web applications.

2. **What are the features of JavaScript?**

Some key features of JavaScript include:

It is a client-side scripting language.

It is interpreted and dynamically typed.

It supports object-oriented programming.

It provides a rich set of built-in functions and libraries.

3. **What are the different data types in JavaScript?**

JavaScript has six primitive data types: string, number, boolean, null, undefined, and symbol. It also has one complex data type, which is object.

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

var has function scope and can be redeclared and reassigned.

let has block scope and can be reassigned but not redeclared.

const has block scope and cannot be redeclared or reassigned.

5. **What is the difference between == and === in JavaScript?**

== performs type coercion and checks for equality after converting the operands to a common type.

=== does not perform type coercion and checks for strict equality without converting the operands.

6. **What is an immediately-invoked function expression (IIFE)?**

An IIFE is a JavaScript function that is immediately executed after it is defined. It is typically used to create a new scope and avoid polluting the global scope.

7. **What is a closure in JavaScript?**

A closure is a combination of a function and the lexical environment within which that function was declared. It allows the function to access variables from its outer scope even after the outer function has finished executing.

8. **What is the purpose of the this keyword in JavaScript?**

The this keyword refers to the object that is currently executing the code. Its value is determined by how a function is invoked.

9. **What is event delegation in JavaScript?**

Event delegation is a technique where you attach a single event listener to a parent element instead of attaching multiple event listeners to individual child elements. The events that occur on the child elements are then handled by the parent element.

10. **Explain the concept of prototypal inheritance in JavaScript.**

In JavaScript, objects can inherit properties and methods from other objects. Each object has an internal link to another object called its prototype. If a property or method is not found on the object itself, JavaScript looks up the prototype chain until it finds the property or method.

11. **What is the difference between null and undefined?**

null represents the intentional absence of any object value.

undefined represents the value assigned to a variable that has been declared but not initialized.

12. **What is hoisting in JavaScript?**

Hoisting is a JavaScript behavior where variable and function declarations are moved to the top of their containing scope during the compilation phase. However, only the declarations are hoisted, not the initializations.

13. **What are the different ways to handle asynchronous operations in JavaScript?**

There are several ways to handle asynchronous operations in JavaScript:

Callback functions

Promises

Async/await

14. **What is a callback function?**

A callback function is a function that is passed as an argument to another function and is executed when a certain event occurs or when the parent function completes its task.

15. **What are promises in JavaScript?**

Promises are objects that represent the eventual completion or failure of an asynchronous operation. They allow you to chain asynchronous operations and handle the success or failure of the operation using .then() and .catch().

16. **What is the purpose of the async keyword in JavaScript?**

The async keyword is used to define an asynchronous function. An asynchronous function returns a promise and allows the use of the await keyword to pause the execution of the function until a promise is resolved or rejected.

17. **What is the difference between a deep copy and a shallow copy?**

A shallow copy creates a new object and copies the references of the original object's properties. If a property is an object, both the original and copied object will refer to the same object.

A deep copy creates a new object and recursively copies all properties of the original object, including nested objects, so that the copied object is completely independent of the original object.

18. **What is event bubbling in JavaScript?**

Event bubbling is a mechanism in which an event triggered on a specific element will "bubble up" and trigger event handlers on its parent elements. This process continues until the event reaches the root element or until an event handler stops the propagation using the stopPropagation() method.

19. **What is event capturing in JavaScript?**

Event capturing is the opposite of event bubbling. In event capturing, the event is first captured by the outermost element and then propagated to the inner elements. It follows the top-down approach.

20. **What is the purpose of the bind() method in JavaScript?**

The bind() method is used to create a new function with a specific this value and, optionally, initial arguments. It allows you to explicitly bind a function to a particular object, which is useful in cases where you want to pass the function as a callback or event handler.

21. **What is the difference between the call() and apply() methods?**

Both call() and apply() are used to invoke a function with a specified this value and arguments. The difference is in how the arguments are passed:

call() accepts the arguments as a comma-separated list.

apply() accepts the arguments as an array or an array-like object.

22. **How do you compare two objects for equality in JavaScript?**

In JavaScript, two objects are considered equal only if they reference the same object in memory. There is no built-in way to compare the contents of two objects for equality. However, you can manually compare the properties of the objects to determine equality.

23. **What is the purpose of the reduce() method in JavaScript?**

The reduce() method is used to reduce an array to a single value by executing a reducer function for each element. It takes an accumulator and the current element as arguments and returns the updated accumulator.

24. **How do you clone an object in JavaScript?**

You can clone an object using various methods, such as using Object.assign(), the spread operator (...), or JSON.parse(JSON.stringify()). However, be aware that deep cloning nested objects and preserving object methods require additional considerations.

25. **Explain the concept of event loop in JavaScript.**

The event loop is a mechanism in JavaScript that handles asynchronous callbacks and ensures that they are executed in a specific order. It continuously checks the call stack for tasks, executes them, and then checks the callback queue for any pending callbacks to be executed.

26. **What are arrow functions in JavaScript?**

Arrow functions are a concise syntax for writing JavaScript functions. They have a shorter syntax compared to regular functions and automatically bind this lexically, based on the surrounding context.

27. **What is the purpose of the map() method in JavaScript?**

The map() method is used to create a new array by applying a transformation function to each element of the original array. It does not modify the original array.

28. **What is the purpose of the filter() method in JavaScript?**

The filter() method is used to create a new array containing only the elements that pass a certain condition. It returns a new array with the elements for which the condition is true.

29. **What is the purpose of the forEach() method in JavaScript?**

The forEach() method is used to execute a provided function once for each element in an array. It does not return a new array.

30. **What is the difference between null and undefined?**

null is an assignment value that represents the intentional absence of any object value.

undefined is a primitive value that is assigned to variables that have been declared but not initialized.

31. **What is the difference between function declarations and function expressions?**

Function declarations are hoisted and can be invoked before they are declared.

Function expressions are not hoisted and must be declared before they are invoked.

32. **How do you handle errors in JavaScript?**

You can handle errors in JavaScript using try...catch blocks. The code that may potentially throw an error is placed inside the try block, and any errors are caught and handled in the catch block.

33. **What is the purpose of the setTimeout() function?**

The setTimeout() function is used to schedule the execution of a function or the evaluation of an expression after a specified delay (in milliseconds). It is often used to create a delay or to execute code asynchronously.

34. **What is a JavaScript event?**

A JavaScript event is an action or occurrence that happens while a webpage is being loaded or interacted with. Examples of events include mouse clicks, key presses, form submissions, and page load.

35. **What is the purpose of the addEventListener() method?**

The addEventListener() method is used to attach an event handler function to an element. It allows you to specify the event type and the function to be executed when the event occurs.

36. **What is the difference between event.preventDefault() and event.stopPropagation()?**

event.preventDefault() is used to prevent the default behavior of an event, such as preventing a form submission or preventing a link from navigating to its URL.

event.stopPropagation() is used to stop the propagation of an event through the event bubbling or capturing phase. It prevents the event from triggering event handlers on parent elements.

37. **What are the different types of error in JavaScript?**

JavaScript has several built-in error types, including:

Error: The base error type.

SyntaxError: Occurs when there is a syntax error in the code.

TypeError: Occurs when a variable or parameter is not of the expected type.

ReferenceError: Occurs when an undefined variable is referenced.

RangeError: Occurs when a numeric value is outside the valid range.

EvalError: Occurs when an error occurs during the eval() function.

38. **What is the purpose of the Object.keys() method?**

The Object.keys() method is used to retrieve an array of all enumerable property names of an object. It returns an array containing the keys of the object's own properties.

39. **What is the purpose of the Object.values() method?**

The Object.values() method is used to retrieve an array of all enumerable property values of an object. It returns an array containing the values of the object's own properties.

40. **What is the purpose of the Object.entries() method?**

The Object.entries() method is used to retrieve an array of all enumerable property key-value pairs of an object. It returns an array containing arrays of key-value pairs

41. **What is the purpose of the isNaN() function?**

The isNaN() function is used to check if a value is NaN (Not-a-Number). It returns true if the value is NaN, and false otherwise.

42. **What is event delegation in JavaScript?**

Event delegation is a technique where you attach a single event listener to a parent element instead of attaching multiple event listeners to individual child elements. The events that occur on the child elements are then handled by the parent element.

43. **What is the purpose of the trim() method in JavaScript?**

The trim() method is used to remove leading and trailing whitespace from a string. It returns a new string with the whitespace removed.

44. **What is the difference between a function declaration and a function expression in JavaScript?**

A function declaration is a statement that declares a function and binds it to a name. It is hoisted and can be invoked before it is declared.

A function expression is an expression that creates a function and can be assigned to a variable. It is not hoisted and must be declared before it is invoked.

45. **What is the difference between == and === in JavaScript?**

== is the loose equality operator and performs type coercion if the operands have different types before making the comparison.

=== is the strict equality operator and does not perform type coercion. It only returns true if the operands have the same value and the same type.

46. What are the different types of loops in JavaScript?

JavaScript has several types of loops:

for loop: Executes a block of code a specified number of times.

while loop: Executes a block of code while a specified condition is true.

do...while loop: Executes a block of code at least once, and then repeats as long as a specified condition is true.

for...in loop: Iterates over the enumerable properties of an object.

for...of loop: Iterates over iterable objects, such as arrays or strings.

47. What is the purpose of the continue statement in JavaScript?

The continue statement is used to skip the current iteration of a loop and continue with the next iteration.

48. What is the purpose of the break statement in JavaScript?

The break statement is used to terminate the execution of a loop or a switch statement. It immediately exits the loop or switch block.

49. What is the purpose of the try...catch statement in JavaScript?

The try...catch statement is used to catch and handle exceptions (errors) in JavaScript code. The code that may throw an exception is placed in the try block, and any exceptions are caught and handled in the catch block.

50. What is the purpose of the finally block in a try...catch statement?

The finally block is used to specify code that will be executed regardless of whether an exception is thrown or caught. It is executed after the try and catch blocks, even if an exception occurs.

51. What is a generator function in JavaScript?

A generator function is a special type of function that can be paused and resumed during its execution. It is defined using the function* syntax and uses the yield keyword to control the flow of execution.

52. What is the purpose of the yield keyword in a generator function?

The yield keyword is used to pause the execution of a generator function and yield a value to the code that is iterating over the generator. It allows for lazy evaluation and the production of a sequence of values over time.

53. What is the difference between a spread operator (...) and the Object.assign() method?

The spread operator (...) is used to create a new array or object by expanding the elements or properties of an existing array or object. It is primarily used for shallow copying.

The Object.assign() method is used to copy the values of all enumerable properties from one or more source objects to a target object. It is used for both shallow and deep copying.

54. What is the purpose of the fetch() function in JavaScript?

The fetch() function is used to make asynchronous HTTP requests and retrieve resources from a server. It returns a Promise that resolves to the response from the server.

55. What is the purpose of the localStorage object in JavaScript?

The localStorage object is used to store key-value pairs locally in a web browser. The data stored in localStorage persists even after the browser window is closed, and can be accessed and modified by JavaScript code.

56. What is the purpose of the sessionStorage object in JavaScript?

The sessionStorage object is used to store key-value pairs locally in a web browser, similar to localStorage. However, the data stored in sessionStorage is specific to a particular session and is cleared when the browser window is closed.

57. What is event delegation in JavaScript?

Event delegation is a technique where you attach a single event listener to a parent element instead of attaching multiple event listeners to individual child elements. The events that occur on the child elements are then handled by the parent element.

58. What is a closure in JavaScript?

A closure is a combination of a function and the lexical environment in which it was declared. It allows the function to access variables and parameters from its outer scope, even after the outer function has finished executing.

59. What is the purpose of the this keyword in JavaScript?

The this keyword refers to the object that is currently executing the code. Its value depends on how a function is called or how an object's method is invoked.

60. What are modules in JavaScript?

Modules in JavaScript are reusable pieces of code that encapsulate related functionality. They help organize and maintain code by keeping variables and functions private within the module, and only exposing a public interface for other code to interact with.

61. **What is the purpose of the export keyword in JavaScript modules?**

The export keyword is used to export variables, functions, or objects from a module, making them available for use in other modules. It allows you to selectively expose parts of the module's internal implementation.

62. **What is the purpose of the import keyword in JavaScript modules?**

The import keyword is used to import variables, functions, or objects from other modules. It allows you to use functionality defined in other modules within the current module.

63. **What is the difference between null and undefined?**

null represents the intentional absence of any object value. It is a value that can be assigned to a variable to indicate the absence of an object.

undefined is a primitive value that is assigned to variables that have been declared but not initialized. It indicates the absence of a value.

64. **What is the difference between a shallow copy and a deep copy?**

A shallow copy creates a new object or array and copies the references of the original object's properties. If a property is an object or array, both the original and copied object/array will refer to the same object/array.

A deep copy creates a new object or array and recursively copies all properties and nested properties of the original object. It creates completely independent copies, so changes to the copied object/array do not affect the original.

65. **What is the purpose of the async and await keywords in JavaScript?**

The async keyword is used to define an asynchronous function, which returns a promise. It allows the use of the await keyword inside the function to pause the execution and wait for the resolution of a promise before continuing.

66. **What are the different ways to handle asynchronous operations in JavaScript?**

There are several ways to handle asynchronous operations in JavaScript:

Callbacks: Functions that are passed as arguments and invoked once an asynchronous operation is complete.

Promises: Objects that represent the eventual completion (or failure) of an asynchronous operation.

Async/await: A syntax that allows writing asynchronous code in a more synchronous style using async functions and the await keyword.

67. **What is the purpose of the Promise object in JavaScript?**

The Promise object represents the eventual completion (or failure) of an asynchronous operation and its resulting value. It allows chaining of asynchronous operations and

provides a consistent way to handle success and error cases.

68. **What are the states of a Promise in JavaScript?**

A Promise in JavaScript can be in one of three states:

Pending: The initial state, neither fulfilled nor rejected.

Fulfilled: The asynchronous operation completed successfully, and the promise is fulfilled with a value.

Rejected: The asynchronous operation encountered an error, and the promise is rejected with a reason/error.

69. **How do you handle errors in asynchronous operations using promises?**

You can handle errors in asynchronous operations using the .catch() method of a Promise. It allows you to specify a callback function that will be executed if the promise is rejected with an error.

70. **What is the purpose of the fetch() function in JavaScript?**

The fetch() function is used to make asynchronous HTTP requests and retrieve resources from a server. It returns a Promise that resolves to the response from the server.

71. **What is the difference between the window object and the document object in JavaScript?**

The window object represents the browser window or frame that contains the document. It provides properties and methods for interacting with the browser.

The document object represents the HTML document loaded in the browser window. It provides access to the elements and content of the document.

72. **What is event bubbling in JavaScript?**

Event bubbling is a phenomenon in which an event triggered on a child element propagates upward through its parent elements in the DOM hierarchy. The event starts from the target element and travels up to the root element, triggering event handlers on each element along the way.

73. **What is event capturing in JavaScript?**

Event capturing is the reverse of event bubbling. It is a mechanism in which an event triggered on a parent element propagates downward through its child elements in the DOM hierarchy, triggering event handlers on each element along the way.

74. **What is the difference between event bubbling and event capturing?**

Event bubbling starts from the target element and propagates upward through the parent elements.

Event capturing starts from the root element and propagates downward through the child elements.

Both mechanisms allow event handlers to be triggered on multiple elements during an event propagation.

75. **What is the purpose of the event.preventDefault() method?**

The event.preventDefault() method is used to prevent the default behavior of an event. For example, it can be used to prevent a form submission or prevent a link from navigating to its URL.

76. **What is event delegation in JavaScript?**

Event delegation is a technique where you attach a single event listener to a parent element instead of attaching multiple event listeners to individual child elements. The events that occur on the child elements are then handled by the parent element

77. **What is the purpose of the this keyword in JavaScript?**

The this keyword refers to the object that is currently executing the code. Its value depends on how a function is called or how an object's method is invoked.

78. **What are closures in JavaScript?**

Closures are functions that have access to variables from an outer (enclosing) function's scope, even after the outer function has finished executing. They "remember" the environment in which they were created.

79. **What is hoisting in JavaScript?**

Hoisting is a JavaScript behavior in which variable and function declarations are moved to the top of their containing scope during the compilation phase, before the code is executed. This allows variables and functions to be used before they are declared.

80. **What is the purpose of the typeof operator in JavaScript?**

The typeof operator is used to determine the type of a value or variable. It returns a string indicating the type of the operand.

81. **What is a callback function in JavaScript?**

A callback function is a function that is passed as an argument to another function and is invoked or called inside that function. It allows for asynchronous and event-driven programming.

82. **What is the difference between a named function and an anonymous function?**

A named function is a function that is declared with a name and can be referenced by that name.

An anonymous function is a function that is declared without a name and is usually assigned to a variable or used as an argument to another function.

83. **What is the purpose of the bind() method in JavaScript?**

The bind() method is used to create a new function that, when called, has its this keyword set to a specified value. It allows you to explicitly bind a function to a particular object or context.

84. **What is the purpose of the call() method in JavaScript?**

The call() method is used to invoke a function with a specified this value and arguments provided individually. It allows you to borrow methods from other objects and explicitly set the this value.

85. **What is the purpose of the apply() method in JavaScript?**

The apply() method is used to invoke a function with a specified this value and arguments provided as an array or array-like object. It allows you to borrow methods from other objects and explicitly set the this value.

86. **What is the purpose of the bind() method in JavaScript?**

The bind() method is used to create a new function that, when called, has its this keyword set to a specified value. It allows you to explicitly bind a function to a particular object or context.

87. **What is the purpose of the call() method in JavaScript?**

The call() method is used to invoke a function with a specified this value and arguments provided individually. It allows you to borrow methods from other objects and explicitly set the this value.

88. **What is the purpose of the apply() method in JavaScript?**

The apply() method is used to invoke a function with a specified this value and arguments provided as an array or array-like object. It allows you to borrow methods from other objects and explicitly set the this value.

89. **What is the purpose of the map() method in JavaScript?**

The map() method is used to create a new array by applying a provided function to each element of an existing array. It transforms the elements of the original array and returns a new array of the same length.

90. **What is the purpose of the filter() method in JavaScript?**

The filter() method is used to create a new array containing elements from an existing array that satisfy a provided condition. It tests each element of the array with a callback function and returns a new array with only the elements that pass the test.

91. **What is the purpose of the reduce() method in JavaScript?**

The reduce() method is used to reduce an array to a single value by applying a provided function to each element of the array. It iterates over the elements and accumulates a result based on the return value of the callback function.

92. **What is the purpose of the forEach() method in JavaScript?**

The forEach() method is used to execute a provided function once for each element in an array. It iterates over the array and applies the callback function to each element, but it does not return a new array.

93. **What are arrow functions in JavaScript?**

Arrow functions are a shorthand syntax for defining functions in JavaScript. They provide a concise and more readable way to write function expressions. Arrow functions have a simplified syntax and automatically bind the this value to the context in which they are defined.

94. **What is 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 provides an easy and readable approach to unpacking values from data structures.

95. **What are template literals in JavaScript?**

Template literals, also known as template strings, are a way to create strings that can contain embedded expressions. They are enclosed by backticks ( ) instead of single or double quotes, and expressions are wrapped in ${ } inside the template string.

96. **What is the purpose of the Object.entries() method in JavaScript?**

The Object.entries() method is used to retrieve an array of enumerable property key-value pairs from an object. It returns an array of arrays, where each inner array contains a property key and its corresponding value.

97. **What are JavaScript promises?**

Promises are objects that represent the eventual completion (or failure) of an asynchronous operation and its resulting value. They are used for handling asynchronous operations in a more readable and manageable way.

98. **What are the states of a JavaScript promise?**

A JavaScript promise can be in one of three states:

Pending: The initial state, neither fulfilled nor rejected.

Fulfilled: The asynchronous operation completed successfully, and the promise is fulfilled with a value.

Rejected: The asynchronous operation encountered an error, and the promise is rejected with a reason/error.

99. **What are JavaScript proxies?**

JavaScript proxies are objects that allow you to intercept and customize fundamental operations on other objects. They provide a way to define custom behavior for operations such as property access, assignment, function invocation, etc.

100. **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 for any pending tasks and processes them one by one. It ensures that JavaScript remains single-threaded while handling non-blocking I/O operations.