JavaScript programming Question & Answers

---

1. What is JavaScript?

JavaScript is a client-side scripting language as well as a server-side scripting language. This scripting language can be written into HTML pages (also could use CSS for styling the pages), and web browsers understand the page.

This scripting language also acts like an object-oriented programming language but not a class-based object-oriented language.

2. Who developed JavaScript, and what is the first name of JavaScript?

JavaScript was created by a Netscape programmer, Brendan Eich.

He developed this new scripting language in just ten days.

At the time of launch, it was initially named Mocha, after which it was known as Live Script and later known as JavaScript.

3. What are the differences between Java and JavaScript?

JavaScript

Java is a complete programming language that can be used for backend coding. JavaScript is a coded program that can be introduced to HTML pages (otherwise known as server-side scripting language).

Java is an object-oriented programming (OOPS) or structured programming languages like C++ or C and .net.JavaScript is a client-side scripting language (not fully OOP).

Java creates applications that run in a virtual machine or browser JavaScript code is run on a browser only.

Java code needs to be compiled JavaScript code is all in text.

Cleancode

4. What are the JavaScript Data Types?

Following are the Data types present in JavaScript:

Basically, there are two types of JavaScript Data types.

Primitive Datatypes

Non- Primitive Datatypes

Primitive Datatypes Non-Primitive Data types

String:-String represents the sequence of characters means the combination of characters, Ex:-'Hello'. Object:-Object represents instances through which we all can access members

Number:-Numbers represents the numeric values, Ex:-2000. Array:-Array represents' a group of similar type data.

Boolean:-Boolean represents the Boolean value, i.e. either true or false. RegExp:-It represents Regular Expression.

Undefined:-it Represents undefined value or not defined values.

Null:-It represents null values means there is no value.

5. Why Should We Study JavaScript?

JavaScript is one of the three languages all web developers must learn for the following reasons:

HTML is used to define the content of web pages. It is otherwise known as the skeleton of web pages.

CSS is used to specify the layout or give styling to the web pages, otherwise known as the shape of the body or cover of the skeleton.
JavaScript to program the behaviour of web pages or web pages workability.

6. What is the basic use of the is NaN function in JavaScript?
The function returns true if the argument is not a number. If the argument is a number, then it returns as false.

7. Who is faster among JavaScript and ASP Script?
JavaScript is faster.
JavaScript is more rapid because, as JS is a client-side language and that it does not need any assistance or help of the webserver to execute, but on the other hand, ASP is a server-side language. That's why ASP is always slower than JavaScript.
JS now is also known as a server-side language named NodeJS.

8. What do you mean by negative infinity?
Negative Infinity is nothing but a number in JavaScript that can be derived by dividing negative numbers by zero. This could be generated by arithmetic operations.

9. Can we break JavaScript Code into several lines? If yes, then How?
Yes, We can break JavaScript code into several lines; we can break within a string statement using a backslash ('\') at the end of the first line code.
For example,
document.write ("This is \a program");
And when you are not within a strong statement and want to change to a new line, then JavaScript ignores the break in the line.
For Example:
var x=1, y=2,
z=x+y;
The above code is perfect for better understanding, but it might hamper our debugging, so it is not advisable to write.

10. What are undeclared and undefined variables?
When variables are not declared in a program, then it is known as Undeclared Variables.
If no variable exists in our program and the program wants to read those variables, it will generate a runtime error.
When there is the declaration of a variable given no value to the variable inside a program is known as an Undefined Variable.
When a program wants to read the variable's value, then the undefined values are returned.
JavaScript Coding Questions

11. Write a code for adding new elements dynamically in JavaScript?

```
<html>
<head>
<title>Elements Dynamically</title>
<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("Dynamic").appendChild(newP);
}
</script>
</head>
<body>
<p id="Dynamic">Dynamic</p>
</body>
</html>
```

This is a simple code for representing how to add new elements dynamically.
So, at first, we set the boilerplate of HTML or the HTML structure. After that, we mention or indicate to DOM that it is a JavaScript file.
After that, we tell a function addNode, then we create an element of p or paragraph type, then we create text node by applying createTextNode (“This is a new text node”); then we will append or assign the text to this p-type.
For execution, we specify <p id=” dynamic”>dynamic</p> then we can process our program by document.getElementbyid (“Dynamic”); and append our p-type then the code will successfully be executed

12. What do you mean by global variables? Define the Declaration and Problem with Global Variable?

The variable which has no scope or available throughout the length of the code is otherwise known as the Global variable.
For declaring a local variable, a var keyword is used. It is also applicable for declaring an object.
When there is the commission of the var keyword, then the global variable is declared.

For Example:

```
// Declaration of a global globalVariable = "Test";
```

13. What is a prompt box?

A prompt is a type of box. It allows the user to enter their input, provide a text box, number, and text provided by label and box.

14. What do you mean by the ‘this’ keyword in JavaScript?

In the case of Java, the ‘This’ keyword is used to point to the current object, but in JavaScript, the ‘This’ keyword refers to them from where it was called.
In other Words, we could say that the” this” keyword refers to the object it belongs to.
So, the ‘this’ keyword has different values according to or depending upon where it is used.
If we use it in a method then, this refers to the owner object.
If it is alone, then this refers to the Global object.
If it is used in a function, then this refers to the global object
If it is used in a function, in strict mode, then this remains undefined.
In an event, this refers to that respective element that receives the event.
Call() and apply() method refer ‘this’ to any object.

15. Explain the working of timers in JavaScript?

Timers are used to execute a bit of code at a set time and repeat the bit of code in a given interval.

Those working are done by using functions setTimeout, setInterval, clear interval.

Working of Functions:

The setTimeout(function, delay) function is used to start a timer, which calls a particular function after the, particularly mentioned delay.

The setInterval(function, delay) function is used to repeatedly execute the given function in the said delay and only stops when it is cancelled.

The clearInterval(id) function instructs or indicates the timer to stop (for stopping the given or mentioned function).

The whole timers are operated within a single thread, and for his events might wait or queue up, they wait for their execution.

16. How do we Define comment's in JavaScript?

For Single line comments:-"//" is used.

For Multi line comments:-"/* is used for multi-line comment */".

17. What will be the output of the below code?

```
var myarray =new Array (1,4,3,6,10,0,22)
document.write(myarray.sort())
myarray.sort(function(a, b) { return b - a; });
document.write(myarray);
```

Ans: [0,1,10,22,3,4,6] and [22,10,6,4,3,1,0]

The sort() method sorts the elements of an array. The sort order can be either alphabetic or numeric, and either ascending (up) or descending (down). By default, the sort() method sorts the values as strings in alphabetical and ascending order.

18. Create a button element with a value attribute set to "CLICK ME and WAIT" and invoke a click event on the button, which calls a function that changes the button's text value color to red. At the same time, this function also sets up a timed function using setTimeout() that sets the text color back to black after 5 seconds.

```
<script type="text/javascript">
function setRed ( )
{
document.getElementById("Button1").style.color = "#FF0000";
setTimeout ( "setBlack()", 5000 );
}
function setBlack ( )
{
document.getElementById("Button1").style.color = "#000000";
}
</script>
<input type="button" name="clickMe" id="Button1" value="Click me and wait!"
onclick="setRed()"/>
```

19. What is the output of the below code?

```
function checkAge(age) {
```

```
 if (age < 18) {
   const message = “Sorry, you’re too young to get your driving license.”;
 }
else {
  const message = “Yay! You’re are eligible!”;
    }
 return message;
}
console.log(checkAge(21));
```

Ans: Reference Error
Variables with the const and let keyword are block-scoped. A block is anything between curly brackets ({ }). In this case, the curly brackets of the if/else statements. You cannot reference a variable outside of the block it’s declared in, a Reference Error gets thrown.

20. Write some JavaScript that uses the current clock and gives an alert message whether the science class is over (Assume class ends at 02:30).
Ans: Possible solution:

```
<script>
var cTime = new Date();
var hour = cTime.getHours();
var mins = cTime.getMinutes();
if (hour > 12 || hour == 12 && mins > 20) {
alert(“Yay, science class got over!!”);
} else {
alert(“Hang on, the science class is yet to finish!”);
}
<script>
```

Cleancode

21. How many alert dialogs will the following Javascript generate, and what will be displayed in each of them?

```
var x = “20”;
function func1(){
var x = “5”;
alert(this.x);
function func2(){alert(x);}
func2();
}
      func1();
```

Ans: A. There will be 2 alert dialogs. The first will display “20”, and the second will display “5”.
JavaScript interview questions for Intermediate

22. Write the Difference between ViewState and SessionState?
ViewState SessionState
ViewState’ is specific to only a page in a session. SessionState’ specific to user data to access all pages inside the web application.
ViewState is only visible from a single page, not from multiple pages. In SessionState, the data availability is access through all of the web pages.

In ViewState, information is stored on the client-side. In SessionState, information is stored on the server.
ViewState value could be lost or cleared when a new page loaded. The programmer himself clears the SessionState value, and in other cases is when the timeouts value gets cleared.

23. What do you mean by the "===" operator?
This operator is called a strict equality operator.
It returns true when the two operands have the same value without having any type of conversion.
So, we could say that it is a strict equality operator which returns false values are similar types.

24. Explain briefly how you can submit a form using JavaScript?
In JavaScript, you can use an on click event for submitting a form, i.e., form.submit() method.
You can perform a submit action by using submit button, By clicking on Hyperlinks.
For submitting a form using JavaScript use document.form[0].submit();
For Example:
document.form[0].submit();

25. Does JavaScript support automatic type conversion (AutoConversion)?
Automatic Type conversion is otherwise known as AutoConversion. When a lower precision data type gets converted to a higher precision data type, it is called typecasting.
For Ex: When we want to convert byte type data to short type data, it could be done easily as a byte is a lower precision data type and a short data type.
ByteShort
Yes, JavaScript supports automatic type conversion; type conversion is the common way followed by JS programmers or Developers.
Automatic Type conversion is otherwise known as "Widening conversion" and "Implicit conversion".
It is the simple concept of assigning lower data types to higher data types.

26. How should we change the style/class of an element?
First of all, we can use the className to assign a value directly to the class. If any such classes are already present on the element, then this will override them.
For getting the value of class on the element, we can add multiple spaces using className.
Given the following way, we can change the class of an element:
document.getElementById("myText").style.fontSize = "20";
or,
document.getElementById ("myText").className = "any class";

27. Explain about Read and Write of a file using JavaScript?
Basically, there are two ways to read and write a file:
Using JavaScript extensions.
Using a web page and Active X objects.
Given are the steps to read and write a file:

Step 1:
file=fopen(getScriptPath(),0);
fread() is used for reading the file content

Step 2:
str = fread(file,flength(file);
The function fwrite() is used to write the contents of the file.

Step 3:
file = fopen("c:\MyFile.txt", 3);// it is used for opening the file for writing.
fwrite(file, str);
// str which is specified within the method is the content that is to be written into the file.

28. What are all the loops available in JavaScript?
The available loops available in JavaScript are given below:
For loop statement:
The JavaScript For loop is the same as for loops of Java and C. For loop continues until a specified condition evaluates to false.
Ex:
for(int i=0; i<=3; i++)
While loop:
A while statement executes its statement until a specified condition evaluates to true, a while statement looks like as follows:
Ex:
while(Condition)
Statement
do-while loops:
The do-while loop continues until a specified condition is false.
Ex: do statement while(Condition)

Cleancode

29. What is called Variable typing in JavaScript?
Variable typing is nothing much firstly, it is used to assign a number to a variable, and after that, the same variable is assigned t a String.

For Example:
i = 10;
i = "string";
So this is called variable typing. This concept of JS is similar to Java.

30. Function's for converting a string of any base class to integer in JavaScript?
The parseInt() function's responsibility is to convert numbers between different bases. parseInt() function takes the string to be converted as its first parameter, and the second parameter is the base of the given string.

31. Explain the difference between the "==" and "===" operator?
" ==" operator checks only for equals of the value whereas,
"===" operator also checks equality of value but in a stricter way and returns false if either the value or the type of variables are different.

32. What Should be the answer of 3+2+ "7"?

As we all know that 3 and 2 are integers or integer types, so they will be added numerically. And after that, since 7 is a string, it doesn't add, and it will be concatenated because the string always is concatenated and not added, So the final result should be 57.

33. What do you understand by "this" keyword in javascript?

Ans: In JavaScript "this" keyword is widely used and it points to a specific object that is executing a current piece of code. It refers to the object that is executing the current function. If the function is a regular function, "this" refers to the global object. If the function referenced is a method in an object, "this" refers to the object itself.

Below are the rules that apply to "this" keyword to know which object it is referencing:

Global Scope

If a function is called from a global scope which includes 'this' keyword, it will always point to the window object.

For Example:

```
<script>
//global declaration of num having global scope
    var num = 150;
    function global_this() {
// local variable inside function has local scope
     var num = 250;
     alert("mynum = " + mynum); // answer is 250 called locally
     alert("this.mynum = " + this.mynum); // answer is 150 as "this" keyword points to //global
variable i.e., window object.
    }
    global_this();
  </script>
```

Object's Method

By using a new keyword if an object is created then "this" will point to that specific object

For example:

```
<script>
// declaring global variables 100 has global scope
    var myNum = 100;
    function printNum() {
     this.myNum = 300;
     this.display = function () {
      var myNum = 400;
      alert("myNum = " + myNum); // 400
      alert("this.myNum = " + this.myNum); // 300
     };
    }
    var objM = new printNum();
    objM.display();
  </script>
```

Cleancode

“this” keyword inside the display() method of the object ‘objM’ will point to the value outside the scope of the display() method.

call() or apply() method

A function can be invoked using the () operator or using call() and apply the () method also. The main aim of call() and apply() is to set the context of “this” keyword inside a function regardless of whether that function is being called in the global scope or as an object’s method.

bind() method

The bind() method is used to set the context of ‘this’ keyword to a particular object when a function is invoked. It is mostly helpful in placing the context of this for a callback function.

For example:

```
var myNum = 50;
function test() {
   alert(this.myNum);
}
var obj_1 = { myNum : 100 , demo1: test};
var obj_2 = { myNum : 150 , demo1:test };
test(); // ‘this’ will point to the global window object
test.call(obj_1); // ‘this’ will point to obj_1
test.apply(obj_2); // ‘this’ will point to obj_2
obj_1.demo1.call(window); // ‘this’ will point to global window object
test.apply(obj_2); // ‘this’ will point to obj_2

<script>

<script>
//global scope of variable myNum
var myNum = 200;
function test(call)
{
//local scope of variable myNum
   var myNum = 250;
   call();
};
var obj1 = {
         myVar: 350,
         Thisfunc : function() {
            alert(“‘this’ points to myNum ” + this + “, myNum = ” + this.myNum);
         }
    };

test(obj1.Thisfunc);  //this points to the global window object
test(obj1.Thisfunc.bind(obj1)); // setting “this” value using bind() method
```

34. What is the result of 3+2+”7”?

Ans: The result is 57. With the + operator 3 and 2 are added. As 7 is a string, the + operator concatenates the expression.

Cleancode

35. List the disadvantages of using innerHTML in javascript.

Ans: The innerHTML property is a part of the DOM and is used to set or return the HTML content of an element. The return value represents the text content of the HTML element. It allows JavaScript code to make changes to a website being rendered.

The disadvantages of using innerHTML are:

Appending to innerHTML is not supported without reparsing the whole innerHTML.

As reparsing is required for innerHTML the processing is slow and takes more time.

The event handlers do not attach automatically to the newly created elements by setting innerHTML. One must keep track of the event handler and attach the new element manually.

By using innerHTML if you add, append, delete or modify contents on a webpage all contents are replaced, also all the DOM nodes inside that element are reparsed and recreated.

No proper validation is provided by innerHTML, hence any valid HTML code can be used. This may break the document of JavaScript.

36. What is a continue statement in javascript?

Ans: The continue statement moves or jumps over the current iteration in the loop if a particular condition occurs, and continues with the execution of the next iteration in the loop.

It can be used in looping statements such as for loop, while loop, and do-while loop. When it is used in a while loop, then it moves back to the condition. If it is used in a for loop, the flow moves to the update expression.

Using the continue statement, the program's flow immediately jumps to the conditional statements, and if the condition is true, then the next iteration will be started; else, the control comes out of the loop.

Cleancode

Syntax:

1

2

(with or without label)

Continue; Or continue label_name

37. What do you understand by event bubbling?

Ans: Event flow specifies the order in which events are received on the page from the element where the event occurs and propagated through the DOM tree. There are two main event models: event bubbling and event capturing.

In the event bubbling model (bottom to top), an event starts at the most particular element and then flows upward toward the least specific element i.e., the document or even the window. For example, you have a div element and a button inside the div element when the button triggers a click event, the click event occurs in the following order:

button

div with the id container

body

html

Document

The click event first occurs on the button, which is the element that was clicked. Then the click event goes up the DOM tree, firing on each node along its way until it reaches the document object. Few web browsers these days will bubble the event up to the window object.

38. Is JavaScript case-sensitive?

Yes, JavaScript is a case-sensitive language. JavaScript also has a set of rules for writing JavaScript programs or codes where the identifiers, variables, keywords, and function names must be written using an appropriate capitalization of letters.

For example:

```
<script>

    var name,Name;

    name='abc';        //variable 1

    Name='def';       // variable 2

   document.write(Name);

</script>
```

In JavaScript, name, and Name are not the same thing even if both variables are spelled the same.

Cleancode

39.What is the difference between a web garden and a web frame?

Web garden Web frame

Web Garden is the web hosting system that encompasses multiple "processes". Web Farm is the web hosting system that encompasses multiple "computers".

It has an application pool(a container of work processes) that can be configured and can define the number of work processes for that pool. multiple web servers are available for multiple clients.

used while hosting multiple processes on a single web server. used while hosting a single web application on multiple web servers in order to distribute the load among the web servers.

40. Describe the role of deferred scripts in javascript.

Ans: The defer Attribute:

The defer attribute tells the browser to continue to process the HTML and build DOM and does not wait till the script file is executed fully. The script loads "in the background", and then runs later when the DOM is completely built.

For example:

```
<script defer src="samplescript.js">
```

The script is loaded asynchronously, the script file can be downloaded while the HTML document is still parsing, even if the file is fully downloaded before the HTML document is finished parsing, the script is not executed until the parsing is complete. Hence, scripts with a defer attribute will never block the page and always execute when the DOM is completely ready.

41. What are the different functional components of JavaScript?

Ans: The basic JavaScript functions are called functional components. These can be created in two ways:

Using function keyword:

```
function functionName (parameters) {
// Code goes here…
}
```

Using function expression or Anonymous function: declare a variable then assign a function it without name.

```
const show= function () {
console.log('Anonymous function!')
}
```

Can also be declared using arrow function

```
const show=()=> {
console.log('Anonymous function!')
}
```

42. What do you mean by screen objects? Explain.

Ans: The screen object is a built-in interface that is used to get the information about the browser screen on which the current webpage is displayed. It provides information about the dimensions of the rendered screen such as screen width, height, colorDepth, pixelDepth, etc.

Property of JavaScript screen object that returns information of the browser:

width: returns the width of the screen
height: returns the height of the screen
availWidth: returns the available width excluding windows taskbar
availHeight: returns the available height excluding windows taskbar
colorDepth: returns the color depth of color palette, in bits, to render images
pixelDepth: returns the color resolution in bits per pixel of the screen.
For example:

```
<script>

document.write("<br/>The display screen width is: "+screen.width);

document.write("<br/>The display screen height is: "+screen.height);

document.write("<br/>The display screen available width is: "+screen.availWidth);

document.write("<br/>The display screen available height is: "+screen.availHeight);

document.write("<br/>The display screen color depth is: "+screen.colorDepth);

document.write("<br/>The display screen pixel depth is: "+screen.pixelDepth);

</script>
```

Cleancode

43. What do you mean by the unshift() method?
Ans: unshift() is a built-in object array method that will add array elements to the front of an array. It overwrites the array original array by adding new array elements to the beginning of an array.

For example:

<script>

const number= [“one”, “two”, “three”, “four”];

number.unshift(“five”, “six”);

document.getElementById(“test”).innerHTML = number;

</script>
Output: five, six, one, two, three, four

44. What are the unescape() and escape() functions in javascript?
Ans: The escape() function in JavaScript is used for encoding (the process of converting plaintext to ciphertext) a string.
Syntax:
1
escape(string to be encoded)
The unescape() function is used to decode(decrypt) that string encoded by the escape() function.
Syntax:
1
unescape(string to be decoded)

45. What do you mean by decodeURI() and encodeURI() in javascript?
Ans: The encodeURI() function encodes the complete URI. It also encodes a few special characters: , / ? : @ & = + $ #
Syntax:
1
encodeURI(complete uri string to be encoded)
The decodeURI() function decodes the URI generated by the encodeURI() function.
Syntax:
1
decodeURI(complete uri string that has been encoded)

46. Are ECMAScript and javascript related? How?
Ans: ECMAScript is a standard for creating a scripting language. Introduced by ECMA International and is basically an implementation with which we learn how to create a scripting language.

Cleancode

Javascript is a general-purpose scripting language that follows the specification of ECMAScript. It is mostly an implementation which tells how to use a scripting language.

47. What do you mean by QuickSort Algorithm in javascript?

Ans: Quicksort algorithm is one of the most popular sorting algorithms in any programming language. QuickSort algorithm follows the divide and conquers method. It divides elements into smaller parts based on several conditions and performs the sort operations on those divided smaller parts. It works well when working with large datasets.

The steps on how the Quicksort algorithm works:

First: select a pivot element.

Second: compare all array elements with the selected pivot element.

Third: arrange them in a way that elements less than the pivot element is to its left and greater than the pivot is at its right.

Finally: execute the same operations on both left and right side elements of the pivot element.

Javascript Logical Questions.

48. How to detect the operating system on the client machine?

if we want to detect the operating system on the client machine, then we have to use navigator.appVersion or navigator.userAgent property.

49. What do you mean by NULL in JavaScript?

The NULL is used to represent no-value or no-object.

It implies there shouldn't be no object or null string, no valid boolean value, no number, and no array object. The value will be nothing, or it means null.

Cleancode

50. What is the use of the delete operator?

The Delete keyword is used for deleting purposes. The delete keyword is used to delete the property as well as its value also.

For Example:

```
var student= {age:20, batch:"ABC"};
delete student.age;
```

51. What do you mean by undefined value in JavaScript?

First of all, an Undefined value means the variable used in the program or code doesn't exist, and another is the value that was not assigned and property doesn't exist, which is known as an Undefined value.

52. Name all the types of Pop up boxes available in JavaScript?

There are several types of pop-box are available in JavaScript.

Alert Box:

An Alert box is used to ensure that the information comes through to the user end. When an alert box pops up, the user has to click the "OK" button for further proceeding.

Syntax:

The window. alert() method is used for pop-ups.

```
1
2
Window.alert("Sai");
```

alert("I am from great learning");
Confirm Box:
A confirm box is used if somebody wants the user to verify or accept something.

When a pop box appears, the user only has to do either click on the “OK” button or click on the “CANCEL” button.

If the user clicks on the “OK” button, then the box returns true, and if the user clicks on the “CANCEL” button, then the box returns false.

Syntax:

```
1
2
3
4
5
6
Window.confirm(“Yes”);
If (confirm(“press a button”)){
Txt=”you pressed ok”;
}else{
Txt=”you pressed cancel”;
}
```

Cleancode

Prompt Box:
A prompt box Is used for the user to input a value before entering into a page.

When a prompt box pops up, the user has to click either on “OK” or “Cancel” to proceed after entering an input value.

if the user clicks on “OK”, then the box returns a true value, neither on the clicking of the “cancel” button box returns null.

Syntax:

```
1
Window.prompt("hello"," hiii");
```

53. Define the use of Void(0) in JavaScript?
The Void(0) is used to prevent/precautionary steps to prevent the page from refreshing, and the passing parameter “zero” is passed while/during calls.

After passing the parameter and calling, Void(0) is used to call another method without refreshing the page.

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

The following code could be chosen to get desired output:

<script language="JavaScript" type="text/JavaScript" >
<!-- location.href="http://newhost/newpath/newfile.html"; //--></script>
56. What is the data type of variables in JavaScript?
All variables in JavaScript are object data types.

54. State the Difference between an Alert Box and a Confirmation Box?
As the name suggests, “alert” gives a pop-up displaying only one button, which is an “OK” button, but in the other case, the Confirmation box displays two buttons that contain one “OK” button and another one is the” CANCEL” button.
This is the fundamental difference between the alert box and the confirmation box.

55. What do you mean by Escape characters?
The Escape Character or backslash is placed or used before characters to make them visible.
The Escape character (Backslash) is used when working with a special type of character like quotes, double quotes, apostrophes, and ampersands.
For Example:
document. write "I m a "good" boy"
document. write "I m a \ "good\" boy"

56. What do you mean by Cookies JavaScript?
Cookies are data, which are stored in small text files on our computer.
We can Say cookies are used to visit websites faster after accepting accept the cookies.
Cookies are the small text files stored in a computer, and cookies get created when the user visits the websites to store information to use it at the time of their need.

Simply, we can say that the cookies concept is introduced for remembering information about the user.

Ex:

username: Great Learning;
60. How should we create a cookie in JavaScript?
By using JavaScript, we can create, read, and delete (crud operations) cookies with the “document. cookie” property.

To create a cookie JavaScript:

document.cookie = "username=John Doe";
You can also add an expiry date for a cookie, but, by default, the expiry date of a cookie is when you close your respective browser, the cookie gets deleted or expired.

57. How can you read a cookie in JavaScript?
By using our JavaScript concept we can read a cookie given below:

var x = document.cookie;
62. Explain what the pop()method in JavaScript is?

The simple working of pop() method to removing the last element from an array and returning that element, by working with this method, also changes the length of the array.

The pop() method is acts similar to the shift() method.

The difference is that is the Shift() method works at the start of the array, and also the pop() method takes the last element of the specified or given array and returns it. After the array is called, it is altered.

For Example:

```
var cars = ["Audi", "BMW", "Mercedes"];
cars.pop();
//Now cars become Audi, BMW.
```

58. Is JavaScript contains concept level scope?

The scope is the context on where the variables for functions can be accessed, as you all write in Java and C, C++. I.e. defined by { }.

The concept level is otherwise known as block-level scope. As JavaScript supports Function-level scope.

So, JavaScript does not have a concept-level scope.

Due to the variables declared inside or within the function have scope inside the function.

59. What are the two primary groups of data types in JavaScript?

The primary groups of data types JavaScript are mentioned below:

Primitive Types

Reference Types

Primitive Types:

Primitive types are number and Boolean type data types.

Reference Types:

Reference types are the more complex types. It is like strings and dates.

60. What is DOM?

DOM stands for Document Object Model.

When a web page is loaded, then the browser creates a Document object model of that page.

The Document Object Model defines a standard or rules for accessing documents on web pages.

61. What is the HTML DOM model?

Simply, we can say that an HTML DOM is nothing but a defining standard for how to get, change, delete, add HTML elements.

The HTML DOM is a standard object model and programming interface for HTML.

62. How can Generic objects be created?

Generic objects can be created as:

```
var I = new object();
```

68. Write the uses of the typeof operator in JavaScript?

The “typeof” operator is used to return a string description of the type of a variable.

Javascript interview questions for Experienced.

63. How to handle exceptions in JavaScript?

The exception is the abnormal termination of a program is called an exception.
An unwanted or unexpected event that disturbs the normal flow of the program is otherwise called an exception.
Exceptions are caused by our program, not by our lack of system resources.
Defining an alternative way to continue the rest of the program is typically called exception handling.
So, we can handle the exception with Try-catch and finally keyword, or we can say that Try-Catch block, finally is used to handle exceptions in JavaScript.
Try-catch block is used to handle the exception, and the 'finally' block is bound to call either their call of try-catch block or not, but finally, the block is bound to call.
In the final block, the programmer can destroy the connection.
Syntax:

```
Try{
//Code
}
Catch(exception){
        //Code to throw an exception
}
Finally{
        //Code runs either it finishes successfully or after catch
}
```

Cleancode

64. Name the keyword which is used to print the text on the screen?

We can write the text on the screen through the document. write keyword.

"document.write("Welcome");"
After that, it will write welcome on the screen.

65. What do you mean by blur function in JavaScript?

The blur event occurs when the element losses its focus or blurriness.
The primary use of the Blur function in a program is to removing focus from a specified object.
Syntax:
In HTML:

```
1
<element onblur=”HTML”>
```
In JavaScript:

```
1
object.onblur= function(){JavaScript};
```

67. What do you mean by variable typing in JavaScript?

Basically, variable typing is used for assigning a number to a variable and after that assigning string to that same variable.

The steps using which you can get more clarifications are given below through an example:

```
i= 8;
i="john";
```

68. How to print Statements in JavaScript?

By using Console.log() function in JavaScript, we can print any variables defined before int, and it is also used to print any message that needs to get displayed to the user.

The Syntax for defining or showing elements to the user is:

```
1
console. log(A);
```

Cleancode

69. How to find an operating system in the client machine using JavaScript?

For finding the Operating system in the client machine using JavaScript using “Navigator. app version” is used.

70. What are the different types of errors are available in JavaScript?

There are three types of errors are in JavaScript:

(1) Load time Errors:

Those errors that come up when we load a web page with inappropriate syntax Errors are otherwise known as Load time Errors. Load time Errors are generated Dynamically.

(2) Run time Errors:

Those Errors which are generated due to misuse of command inside an HTML language are known as Run time errors.

(3) Logical Errors:

Those errors occur due to the formation or writing off bad logic inside the program, which logic has different operations known as Logical errors.

The primary benefit of the push method is that it always adds or appends one or more than one element to the end of an array, so using this method, we can append multiple elements bypassing multiple arguments.

Syntax:

```
1
array.push(item1,item2,……,item10);
```

For Ex:

Var fruits= ["Banana", "apple", "orange"];
fruits.push ("grapes");
Output:

Banana, apple, orange, grapes.

71. What do you mean by unshift method in JavaScript?
Unshift method work is similar to the push method but pushes method to append the elements and unshift method just prepend the elements.
unshift method works beginning of the array. This method is used to prepend one or more than one element at the beginning of an array.

Syntax:
1
array.unshift (item1, item2,….., Item10);
For Ex:

Var fruits= ["Banana", "apple", "orange"];
fruits.unshift ("Grapes");
Output:
Grapes, Banana, apple, orange.

Cleancode

72. What is the difference between JavaScript and Jscript?
The difference between these two scripts is said to be no different.
Both are pretty similar, but the only difference is JavaScript ids developed by Netscape and Jscript is developed by Microsoft.

73. How are the object properties assigned in JavaScript?
You can define a property by assigning its value.
As all of you know, all JavaScript variables like objects names and property names are case-sensitive so that we can assign value through their name and properties.
We can assign object Properties in the following ways:

obj["class"] = 12;
Or,

obj.class = 12;
80. What do you mean by ‘Strict’ mode in JavaScript, and how can it be enabled?
Strict mode is used to solve some of the mistakes that hamper the JavaScript engines for work efficiently,

The strict mode adds a specific constraint to JavaScript, so under the Script mode, JavaScript show errors from few codes.

For Enabling the Strict mode, you have to add the string literal “use strict” above the specified file. You can get a clear idea by the following example:

```
function myfunction() {
        "use strict";
        var v = "This is a strict mode function";
}
```

74. Write the various ways to get the status of a Checkbox?

The status for getting status of a checkbox as follows:

```
alert(document.getElementById('checkbox1').checked);
```

If the Checkbox is checked, then the alert will return TRUE.

75. Explain the window.onload and onDocumentReady perform in JavaScript?

.onload:

The Window.onload will execute the codes when the browser loads the DOM tree and other resources like images and objects.

The .onload function is not run further & until all the information on the page is loaded, leading to some delay before any code is executed.

.onDocumentReady:

The .onDocumentReady is executed when the DOM is load, without waiting for the resources to load like .onload works.

This leads to the .onDocumentReady allows to executes the code faster in DOM.

In the case of .onDocumentReady, it loads the codes just after the Document Object Manipulation is loaded, so this allows early manipulation of the code.

Cleancode

78. What do you mean by closures in JavaScript? When are they used?

The closure is nothing but a locally declared variable or otherwise known as the local variable, which is related to a function, and it stays in memory at the time when the function has returned.

For Example:

```
function greet(message) {
        console.log(message);
        }
        function greeter(name, age) {
         return name + " says howdy!! He is " + age + " years old";
        }

// For Generating the message:-
var message = greeter("James", 23);

// Pass it Explicitly to greet:-
greet(message);
```

So the above Function could be better represented by using closures in this function:

```
{
        var message = name + " says howdy!! He is " + age + " years old";
          return function greet()
```

```
        {
            console.log(message);
        };
}

// For Generating the closures:-
var JamesGreeter = greeter("James", 23);

// For Using the closures:-
JamesGreeter();
```

79. What do you mean by anonymous function in JavaScript? Describe its properties.

A function said to be an anonymous function when the function is declared without any named identifier.

In other words, we can say that the anonymous function is inaccessible after their declaration in a program or a code.

Anonymous function declaration:

```
var anon = function()
        {
        alert('I am anonymous');
        };
anon();
```

80. Explain Hoisting in JavaScript.

Hoisting is a JavaScript behaviour where anywhere the variable/functions are declared they are moved to the top of the scope. Note the scope may be local or global.

Ex: Before Hoisting

```
a = 1;

alert ( ‘ a = ‘  + a);

var a;
```

Ex: After Hoisting

```
var a;

a = 1;

alter( ‘ a = ‘ + a);
```

81. Explain Implicit Type Coercion in JavaScript.

Automatic value conversion from one data type to another data type is known as Implicit type coercion in JavaScript. There are many types Number to String, String to Number, Boolean to Number etc. When different values are applied to the datatypes that are not of that value it automatically converts them.

String Coercion:

The time when the number is added to a string, it converts the number type as a string type with the help of the “+” operator.

Example:

```
var a = 4;
var b = “5”; //number that’s inside double quotes indicate it’s been considered as a string
```

a + b // Returns “45”
Example:
var p = 45;
var q = “Hi”;// Here the number 45 is been concatenated with the string
p + q // Returns “45Hi”;
NOTE: We can use the “ – “ operator as well, but the number which we give will be converted to string datatype and subtracted
Example:
var p = 5;
var q = “5”;
p – q //Returns 0 since the variable q (string type) is converted to a number type

Boolean Coercion
When we use logical operators, if statements, ternary operators and loop checks Boolean coercion takes place. We have to ensure properly the truthy and falsy values.
Except 0, O(n), -0, “”, Null, undefined and Nan all others are truthy values
Example:
var a = 0;
var b = 24; // variable declaration with value
if(a) { console.log(a) } // Since the a value is 0 the code will not run (Falsy)
if(b) { console.log(b) } // Since the b value is 24 the code will run(Truthy)
Logical Operators:
Comparing the logical operators in the other programming languages, in JavaScript, the Logical operators does not return true or false, one of the operands is returned.

Cleancode

OR (| |) operator – The first value is returned only when it is truthy else the second value gets returned.

AND (&&) operator – When both the values are truthy it returns always the second value. If the first value false it returns the first value. If the second value false it will return the second value.

Operator	First Value	Second Value	Returns
OR (| |)	Truthy	Falsy	First Value
Falsy	Truthy	Second Value	
Falsy	Falsy	Second Value	
Truthy	Truthy	First Value	
AND (& &)	Truthy	Falsy	Second value
Falsy	Truthy	First Value	
Truthy	Truthy	Second Value	

Example:
var a = 260;
var b = “Hi”;
var c = NaN;
a | | b // Returns 260 since the first value is truthy
a | | c // Returns 260 since the first value is truthy
a && b // Returns “Hi” since both the values are truthy
b && c // Returns NaN since the second value is falsy

```
if( a && b )
{
  console.log(“Code runs” ); // This block runs because a && b returns “Hi” (Truthy)
}
if( a || c )
{
  console.log(“Code runs”);  // This block runs because a || b returns 260(Truthy)
}
```

Equality Coercion

The operator we use for Equality coercion is “ ==”. It compares only values not datatypes. If there exist two different datatypes then it converts both of them to one type and compares the value.

Example:

```
var a = 12;
var b = “12”;
a == b   //operands are equal since it converts both the datatypes to same and compares
```

82. Is JavaScript a statically typed or a dynamically typed language?

JavaScript is categorized as a dynamically typed language. Because the variable type is checked during runtime parallelly with statically typed language, where the types of a variable are checked during the compilation phase.

Static Typing	Dynamic Typing
string namename = “Peter”;name = 34;	var namename = “Peter”;name = 35;
Variables have types	Variables have no types
Values have types	Values have no types
Variables cannot change types	Variables can change types

This is one of the important Javascript interview questions asked in interviews.

83. Explain passed by value and passed by reference.

JavaScript provides two different categories of datatypes Primitive and Objects

Primitive Datatypes: Number, Boolean, String, Null and Undefined.

Objects: Arrays, functions, plain objects and more Anything except primitive are objects.

NOTE: All primitive data types in JavaScript is passed by value

Pass by value:

In this, the function is called by passing the value directly as an argument. So, any changes that’s made inside a function won’t affect the actual value.

The parameters passed as arguments are mutated (Creation of own copy). So, any changes made inside the function is made to a copied value but not for the original one.

Example:

```
<script>
  let a = 1;
  let change = (val) =>
 {
    val = 2
```

```
  }
  change(a);
  document.write(a);
</script>
```

In the above example variable, a is been assigned with the value 1 and then changed to 2 inside the function called change. Since JavaScript pass by value the output will be 1.

Pass by Reference:

In some instances, there arises a situation, the address is passed instead of arguments to call a function. During that time the value gets changed inside a function affects the variable passed outside the function. This is called a pass by reference. In JavaScript, mostly arrays and objects follow pass by reference.

In the following example, an object named 'b' is declared outside the function 'change'. Here one should heed that variable 'b' got mutated but not assigned with value 2, as shown in example 2. A pass by reference takes place when a mutation has occurred.

Example 1:

```
<script>
  let b = {num:1};
  let change = (val) =>
{
    val.num = 2
  }
  change(b);
 document.write(JSON.stringify(b));
</script>
```

Cleancode

Example 2:

```
<script>
  let b = {num : 1};
  let change = (val) => {
    val = {num :2};

  }
  change(b);
 document.write(JSON.stringify(b));
```

84. What is an Immediately Invoked Function in JavaScript?

Ans:The function which runs as soon as its defined is known as the Immediately Invoked Function in JavaScript.

Syntax:

```
function ()
{
 // Do something;
}) ();
```

85. Explain Higher-Order Functions in JavaScript

Where starting a function with the bracket is necessary else it will be considered as the normal function. The second set of parentheses is used to invoke the function because functions do not work without invoking.

Just like data types such as Number, Boolean, String is considered to be the data then functions can also work as data.
Functions can be passed through other functions
Functions can be set as object properties
Functions can be stored in arrays
Functions can be set as variables

Example:

```
function higher-order(fn)
{
  fn();
}
higherOrder(function() { console.log("Hi") });
```

86. Explain call( ), apply( ) and, bind( ) methods.

Ans:

call( )

It is a library method available in the JavaScript

This method invokes functions/methods by mentioning the owner object

Example:

```
function sayHi()
{
  return "Hi" + this.name;
}
var obj = {name: "Shiv"};
sayHi.call(obj); // Returns "Hi Shiv"
```

This method allows using the method (function) of another object

apply( )

This method is the same as call( ) but apply ( ) takes arguments in the form of array whereas call () takes arguments separately

Example

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

bind ( )

This method uses the "this" keyword which will be bound to the parent object which is considered as a parameter. This method will always return a new function

Example:

```
var bikeDetails = {
   displayDetails: function(registrationNumber,brandName){
   return this.name+ " , "+ "bike details: "+ registrationNumber + " , " + brandName;
  }
```

Cleancode

```
}
var person1 = {name:  “Shiv”};
var detailsOfPerson1 = bikeDetails.displayDetails.bind(person1, “KA020612”, “Dio”);
// Binds the displayDetails function to the person1 object
detailsOfPerson1();
// Returns Shiv, bike details: KA020612, Dio
```

87. What is Currying in JavaScript?

The transformation of functions from f(a,b,c) callable to f(a),(b),(c) callable is known as Currying in JavaScript. This is an advanced technique used to work with functions not only in JavaScript yet in other languages as well.

Example:

```
function curry(f) { // curry(f) does the currying transform
return function(a) {
   return function(b) {
     return f(a, b);
   };
 };
}
```

For Example, if we have a function f(a,b), then the function after currying, will be transformed to f(a)(b).

This is one of the important Javascript interview questions asked in interviews.

88. Explain Scope and Scope Chain in JavaScript.

JavaScript variables are also having scope as other programming languages. The accessibility and visibility of the variables are known as Scope.

There are three types of scopes in JS:

Global Scope:

Variables that are not inside any function or the curly braces is known as Global Scope. These variables can be accessed from all parts of the code.

Local or Function Scope:

Variables declared inside a scope or function is known as function. Variables where it can be accessed only within that function. That means they cannot be accessed outside code.

Block Scope:

The variables in blocked scope are limited only to that particular block within the curly braces mentioned.

Scope Chain: The currently accessible scopes in a code are known as Scope chains. Irrespective of the scopes either it might be under global, local or block. When the JavaScript engine searches for a scope currently, the accessible scopes are termed to be scope chains.

89. What are object prototypes?

The mechanism in which the objects inherit features from one another is known as an object prototype.

Example: (only for understanding purpose non-technical)

Let’s talk about a car guess what the features are, every car has an engine, a staring and 4 wheels. It might be brand X, Y, or Z every car moves using the same mechanism.

So, relating it to object prototype. Brand X is an object same prototype of a basic mechanism is inherited by brand Y with exterior changes and added features. So, this process is known as object prototyping.

90. What are callbacks?

A callback is a function passed as an argument to another function. It acts as a sequencing system for function execution. Once the function is been executed using call back, we can wait for the result and then execute the next function in the sequence.

Ex. setTimeout ( ) timeout method is usually used to cover up the amount of time taken to execute the program.

This is one of the important Javascript interview questions asked in interviews.

91. What is memoization?

The optimization technique which speeds up the applications by storing the results needed to an immediate function calls and returning to the cached result when the same inputs are supplied again is known as memorization

Example:

```
function memoizedAddTo256() //function declaration
{
 var cache = {}; // variable declaration  as cache
 return function(num){
  if(num in cache){
   console.log("cached value"); // if condition is true it returns cached value
   return cache[num] /
  }
  else{
   cache[num] = num + 256;

   return cache[num];
  }
 }
}
var memoizedFunc = memoizedAddTo256();
memoizedFunc(20); // Normal return
memoizedFunc(20); // Cached return
```

92. What is recursion in a programming language?

Ans: Recursion is a technique in which the function calls itself again and again repeatedly until the condition gets false.

Example:

```
function countDown(number) {
  // number is displayed
  console.log(number);
  // value of the number gets decreased in each iteration
  const newNumber = number – 1;
  // condition is passed until its false
  if (newNumber > 0) {
```

Cleancode

```
    countDown(newNumber);
  }
}
countDown(4);
```
Output: 4 3 2 1

93. What is the use of a constructor function in JavaScript?

Ans: A function that creates an instance of a class which is called an object is known as a constructor. Whenever the object is using a new keyword then the constructor gets called. Constructor is used to creating an object and set values if there are any object properties present.

94. What are arrow functions?

Ans:The function which allows declaring shorter syntax which was introduced during the ES6 version is known as the arrow function.

Example: Before

```
hello = function ()
 {
  return “Hi!”;
}
```
After:

```
hello = () =>
{
  return “Hi!”;
}
```
This is one of the important Javascript interview questions asked in interviews.

Cleancode

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

Ans:var is having scope only within the function

let and const are having scope within their blocks between the curly braces

Also, any variable with the keyword const cannot be changed or modified it remains constant

keyword	const	let	var
block scope	yes	yes	no
function scope	yes	yes	yes
global scope	no	no	yes
can be reassigned	no	yes	yes

100. What is Node.js? Where can you use it?

Node.js is an open-source, cross-platform JavaScript runtime environment and library to run web applications outside the client’s browser. It is used to create server-side web applications.

Node.js is perfect for data-intensive applications as it uses an asynchronous, event-driven model. You can use I/O intensive web applications like video streaming sites. You can also use it for developing: Real-time web applications, Network applications, General-purpose applications, and Distributed systems.

101. Why use Node.js?

Node.js makes building scalable network programs easy. Some of its advantages include:
It is generally fast
It rarely blocks
It offers a unified programming language and data type
Everything is asynchronous
It yields great concurrency

102. How does Node.js work?
A web server using Node.js typically has a workflow that is quite similar to the diagram illustrated below. Let's explore this flow of operations in detail.
Node.js Architecture Workflow
Clients send requests to the webserver to interact with the web application. Requests can be non-blocking or blocking:
Querying for data
Deleting data
Updating the data
Node.js retrieves the incoming requests and adds those to the Event Queue
The requests are then passed one-by-one through the Event Loop. It checks if the requests are simple enough not to require any external resources
The Event Loop processes simple requests (non-blocking operations), such as I/O Polling, and returns the responses to the corresponding clients
A single thread from the Thread Pool is assigned to a single complex request. This thread is responsible for completing a particular blocking request by accessing external resources, such as computation, database, file system, etc.
Once the task is carried out completely, the response is sent to the Event Loop that sends that response back to the client.
Basics to Advanced - Learn It All!
Caltech PGP Full Stack DevelopmentEXPLORE PROGRAMBasics to Advanced - Learn It All!

103. Why is Node.js Single-threaded?
Node.js is single-threaded for async processing. By doing async processing on a single-thread under typical web loads, more performance and scalability can be achieved instead of the typical thread-based implementation.

104. If Node.js is single-threaded, then how does it handle concurrency?
The Multi-Threaded Request/Response Stateless Model is not followed by the Node JS Platform, and it adheres to the Single-Threaded Event Loop Model. The Node JS Processing paradigm is heavily influenced by the JavaScript Event-based model and the JavaScript callback system. As a result, Node.js can easily manage more concurrent client requests. The event loop is the processing model's beating heart in Node.js.

105. Explain callback in Node.js.
A callback function is called after a given task. It allows other code to be run in the meantime and prevents any blocking. Being an asynchronous platform, Node.js heavily relies on callback. All APIs of Node are written to support callbacks.

106. What are the advantages of using promises instead of callbacks?
The control flow of asynchronous logic is more specified and structured.

The coupling is low.
We've built-in error handling.
Improved readability.

107. How would you define the term I/O?
The term I/O is used to describe any program, operation, or device that transfers data to or from a medium and to or from another medium
Every transfer is an output from one medium and an input into another. The medium can be a physical device, network, or files within a system
io

108. How is Node.js most frequently used?
Node.js is widely used in the following applications:
Real-time chats
Internet of Things
Complex SPAs (Single-Page Applications)
Real-time collaboration tools
Streaming applications
Microservices architecture

109. Explain the difference between frontend and backend development?
Front-end
Back-end
Frontend refers to the client-side of an application
Backend refers to the server-side of an application
It is the part of a web application that users can see and interact with
It constitutes everything that happens behind the scenes
It typically includes everything that attributes to the visual aspects of a web application
It generally includes a web server that communicates with a database to serve requests
HTML, CSS, JavaScript, AngularJS, and ReactJS are some of the essentials of frontend development
Java, PHP, Python, and Node.js are some of the backend development technologies
Basics to Advanced - Learn It All!
Caltech PGP Full Stack DevelopmentEXPLORE PROGRAMBasics to Advanced - Learn It All!
If you are curious to explore interview questions related to frontend development, you can check out our article on ReactJS Interview Questions and Answers.

110. What is NPM?
NPM stands for Node Package Manager, responsible for managing all the packages and modules for Node.js.
Node Package Manager provides two main functionalities:
Provides online repositories for node.js packages/modules, which are searchable on search.nodejs.org
Provides command-line utility to install Node.js packages and also manages Node.js versions and dependencies

111. What are the modules in Node.js?

Modules are like JavaScript libraries that can be used in a Node.js application to include a set of functions. To include a module in a Node.js application, use the require() function with the parentheses containing the module's name.

112. What is the purpose of the module .Exports?

In Node.js, a module encapsulates all related codes into a single unit of code that can be parsed by moving all relevant functions into a single file. You may export a module with the module and export the function, which lets it be imported into another file with a needed keyword.

113. Why is Node.js preferred over other backend technologies like Java and PHP?

Some of the reasons why Node.js is preferred include:

Node.js is very fast

Node Package Manager has over 50,000 bundles available at the developer's disposal

Perfect for data-intensive, real-time web applications, as Node.js never waits for an API to return data

Better synchronization of code between server and client due to same code base

Easy for web developers to start using Node.js in their projects as it is a JavaScript library

114. What is the difference between Angular and Node.js?

Angular	Node.js
It is a frontend development framework	It is a server-side environment
It is written in TypeScript	It is written in C, C++ languages
Used for building single-page, client-side web applications	Used for building fast and scalable server-side networking applications
Splits a web application into MVC components	Generates database queries

Also Read: What is Angular?

115. Which database is more popularly used with Node.js?

MongoDB is the most common database used with Node.js. It is a NoSQL, cross-platform, document-oriented database that provides high performance, high availability, and easy scalability.

Designing a Social Media App Like Instagram
Free Webinar | Watch the WebcastEXPLORE NOWDesigning a Social Media App Like Instagram

17. What are some of the most commonly used libraries in Node.js?

There are two commonly used libraries in Node.js:

ExpressJS - Express is a flexible Node.js web application framework that provides a wide set of features to develop web and mobile applications.

Mongoose - Mongoose is also a Node.js web application framework that makes it easy to connect an application to a database.

18. What are the pros and cons of Node.js?

NODE Pro	
Fast processing and an event-based model	
Not suitable for heavy computational tasks	

Uses JavaScript, which is well-known amongst Using callback is complex since you end up with several nested callbacks NODE cone	Node Package Manager has over 50,000 packages that provide the functionality to an application Dealing with relational databases is not a good option for Node.js Best suited for streaming huge amounts of data and I/O intensive operations Since Node.js is single-threaded, CPU intensive tasks are not its strong suit

116. What is the command used to import external libraries?

The “require” command is used for importing external libraries. For example - “var http=require (“HTTP”).” This will load the HTTP library and the single exported object through the HTTP variable.
Now that we have covered some of the important beginner-level Node.js interview questions let us look at some of the intermediate-level Node.js interview questions.
varhttp
Node.js Interview Questions and Answers For Intermediate Level

117. What does event-driven programming mean?

An event-driven programming approach uses events to trigger various functions. An event can be anything, such as typing a key or clicking a mouse button. A call-back function is already registered with the element executes whenever an event is triggered.

Cleancode

118. What is an Event Loop in Node.js?

Event loops handle asynchronous callbacks in Node.js. It is the foundation of the non-blocking input/output in Node.js, making it one of the most important environmental features.

119. Differentiate between process.nextTick() and setImmediate()?

The distinction between method and product. This is accomplished through the use of nextTick() and setImmediate(). next Tick() postpones the execution of action until the next pass around the event loop, or it simply calls the callback function once the event loop's current execution is complete, whereas setImmediate() executes a callback on the next cycle of the event loop and returns control to the event loop for any I/O operations.
Here's How to Land a Top Software Developer Job
Full Stack Development-MEANEXPLORE PROGRAMHere's How to Land a Top Software Developer Job

120. What is an EventEmitter in Node.js?

EventEmitter is a class that holds all the objects that can emit events
Whenever an object from the EventEmitter class throws an event, all attached functions are called upon synchronously
/eventemitter

121. What are the two types of API functions in Node.js?

The two types of API functions in Node.js are:

Asynchronous, non-blocking functions
Synchronous, blocking functions

122. What is the package.json file?

The package.json file is the heart of a Node.js system. This file holds the metadata for a particular project. The package.json file is found in the root directory of any Node application or module
This is what a package.json file looks like immediately after creating a Node.js project using the command: npm init
You can edit the parameters when you create a Node.js project.
node-npm

123. How would you use a URL module in Node.js?

Basics to Advanced - Learn It All!
Caltech PGP Full Stack DevelopmentEXPLORE PROGRAMBasics to Advanced - Learn It All!
The URL module in Node.js provides various utilities for URL resolution and parsing. It is a built-in module that helps split up the web address into a readable format.
varurl

124. What is the Express.js package?

Express is a flexible Node.js web application framework that provides a wide set of features to develop both web and mobile applications

Cleancode

125. How do you create a simple Express.js application?

The request object represents the HTTP request and has properties for the request query string, parameters, body, HTTP headers, and so on
The response object represents the HTTP response that an Express app sends when it receives an HTTP request

126. What are streams in Node.js?

Streams are objects that enable you to read data or write data continuously.
There are four types of streams:
Readable – Used for reading operations
Writable – Used for write operations
Duplex – Can be used for both reading and write operations
Transform – A type of duplex stream where the output is computed based on input

127. How do you create a simple server in Node.js that returns Hello World?

simple-server
Import the HTTP module
Use createServer function with a callback function using request and response as parameters.
Type “hello world."
Set the server to listen to port 8080 and assign an IP address
Get the Coding Skills You Need to Succeed
Full Stack Development-MEANEXPLORE PROGRAMGet the Coding Skills You Need to Succeed

128. Explain asynchronous and non-blocking APIs in Node.js.

All Node.js library APIs are asynchronous, which means they are also non-blocking

A Node.js-based server never waits for an API to return data. Instead, it moves to the next API after calling it, and a notification mechanism from a Node.js event responds to the server for the previous API call.

130. How do we implement async in Node.js?

As shown below, the async code asks the JavaScript engine running the code to wait for the request.get() function to complete before moving on to the next line for execution.

async

131. What is a callback function in Node.js?

A callback is a function called after a given task. This prevents any blocking and enables other code to run in the meantime.

In the last section, we will now cover some of the advanced-level Node.js interview questions.

Node.js Interview Questions and Answers For Experienced Professionals

This section will provide you with the Advanced Node.js interview questions which will primarily help experienced professionals.

132. What is REPL in Node.js?

REPL stands for Read Eval Print Loop, and it represents a computer environment. It's similar to a Windows console or Unix/Linux shell in which a command is entered. Then, the system responds with an output

Cleancode

133. What is the control flow function?

The control flow function is a piece of code that runs in between several asynchronous function calls.

134. What is the difference between fork() and spawn() methods in Node.js?

fork()

spawn()

spawn

fork() is a particular case of spawn() that generates a new instance of a V8 engine.

Spawn() launches a new process with the available set of commands.

Multiple workers run on a single node code base for multiple tasks.

This method doesn't generate a new V8 instance, and only a single copy of the node module is active on the processor.

135. What is the buffer class in Node.js?

Buffer class stores raw data similar to an array of integers but corresponds to a raw memory allocation outside the V8 heap. Buffer class is used because pure JavaScript is not compatible with binary data

136. What is piping in Node.js?

Piping is a mechanism used to connect the output of one stream to another stream. It is normally used to retrieve data from one stream and pass output to another stream

137. What is callback hell?
Callback hell, also known as the pyramid of doom, is the result of intensively nested, unreadable, and unmanageable callbacks, which in turn makes the code harder to read and debug
improper implementation of the asynchronous logic causes callback hell

138. What is a reactor pattern in Node.js?
A reactor pattern is a concept of non-blocking I/O operations. This pattern provides a handler that is associated with each I/O operation. As soon as an I/O request is generated, it is then submitted to a demultiplexer

139. For Node.js, why does Google use the V8 engine?
The V8 engine, developed by Google, is open-source and written in C++. Google Chrome makes use of this engine. V8, unlike the other engines, is also utilized for the popular Node.js runtime. V8 was initially intended to improve the speed of JavaScript execution within web browsers. Instead of employing an interpreter, V8 converts JavaScript code into more efficient machine code to increase performance. It turns JavaScript code into machine code during execution by utilizing a JIT (Just-In-Time) compiler, as do many current JavaScript engines such as SpiderMonkey or Rhino (Mozilla).

Cleancode

140. Explain the concept of middleware in Node.js.
Middleware is a function that receives the request and response objects. Most tasks that the middleware functions perform are:
Execute any code
Update or modify the request and the response objects
Finish the request-response cycle
Invoke the next middleware in the stack

141. What are the different types of HTTP requests?
HTTP defines a set of request methods used to perform desired actions. The request methods include:
GET: Used to retrieve the data
POST: Generally used to make a change in state or reactions on the server
HEAD: Similar to the GET method, but asks for the response without the response body
Here's How to Land a Top Software Developer Job
Full Stack Development-MEANEXPLORE PROGRAMHere's How to Land a Top Software Developer Job
DELETE: Used to delete the predetermined resource

142. How would you connect a MongoDB database to Node.js?
To create a database in MongoDB:
Start by creating a MongoClient object

Specify a connection URL with the correct IP address and the name of the database you want to create
varmongo

143. List the various Node.js timing features.
As you prepare for your upcoming job interview, we hope that this comprehensive guide has provided more insight into what types of questions you'll be asked.
timers

145. What is WASI, and why is it being introduced?
The WASI class implements the WASI system called API and extra convenience methods for interacting with WASI-based applications. Every WASI instance represents a unique sandbox environment. Each WASI instance must specify its command-line parameters, environment variables, and sandbox directory structure for security reasons.

146. What is a first-class function in Javascript?
First-class functions are a powerful feature of JavaScript that allows you to write more flexible and reusable code. In Node.js, first-class functions are used extensively in asynchronous programming to write non-blocking code.

147. How do you manage packages in your Node.Js project?
Managing packages in your Node.js project is done using the Node Package Manager (NPM), which allows you to install and manage third-party packages and create and publish your packages.

Cleancode

148. How is Node.js better than other frameworks?
Node.js is a server-side JavaScript runtime environment built on top of the V8 JavaScript engine, the same engine that powers Google Chrome. It makes Node.js very fast and efficient, as well as highly scalable.

149. What is a fork in node JS?
The Fork method in Node.js creates a new child process that runs a separate Node.js instance and can be useful for running CPU-intensive tasks or creating a cluster of Node.js servers.

150. List down the two arguments that async. First, does the queue take as input?
The async.queue function in Node.js takes two arguments as input: a worker function and an optional concurrency limit. It is used to create a task queue executed in parallel.
Here's How to Land a Top Software Developer Job
Full Stack Development-MEANEXPLORE PROGRAMHere's How to Land a Top Software Developer Job

151. What is the purpose of the module.exports?
The module. exports object in Node.js is used to export functions, objects, or values from a module and is returned as the value of the require() function when another module requires a module.

152. What tools can be used to assure consistent code style?

In summary, several tools can be used in Node.js to ensure consistent code style and improve code quality, including ESLint, Prettier, and Jest.

153. What is the difference between JavaScript and Node.js?
Node.js is a runtime environment for executing JavaScript code outside of a web browser, while JavaScript is a programming language that can be executed in both web browsers and Node.js environments.

154. What is the difference between asynchronous and synchronous functions?
Synchronous functions block the execution of other code until they are complete, while asynchronous functions allow other code to continue executing while they are running, making them essential for writing scalable Node.js applications.

155. What are the asynchronous tasks that should occur in an event loop?
Asynchronous tasks that should occur in an event loop in Node.js include I/O operations, timers, and callback functions. By performing these tasks asynchronously, Node.js can handle a large number of concurrent requests without blocking the event loop.

156. What is the order of execution in control flow statements?
In Node.js, control flow statements are executed in a specific order. The order of execution is determined by the event loop. The event loop is a mechanism in Node.js that allows for the execution of non-blocking I/O operations.

Cleancode

157. What are the input arguments for an asynchronous queue?
An asynchronous queue in Node.js is a data structure that allows for the execution of functions in a specific order. Functions are added to the queue and are executed in the order that they were added. An asynchronous queue is useful when you want to execute a series of functions in a specific order.

158. Are there any disadvantages to using Node.js?
Node.Js is not suitable for CPU-intensive tasks. This is because Node.js is single-threaded, meaning it can only execute one task at a time. Node.js is not suitable for applications that require a lot of memory. This is because Node.js uses a lot of memory for each connection. If you have a large number of connections, it can quickly consume a lot of memory.

159. What is the primary reason for using the event-based model in Node.js?
The main reason to use the event-based model in Node.js is performance. The event-based model allows for non-blocking I/O operations, which means that Node.js can handle a large number of connections without using a lot of resources.

160. What is the difference between Node.js and Ajax?
Ajax and Node.js are two different technologies that are used for different purposes. Ajax is a client-side technology that allows for asynchronous communication between the client and the server. It is typically used to update parts of a web page without requiring a full page reload.

Node.js, on the other hand, is a server-side technology that is used for building fast, scalable, and efficient server-side applications. It is typically used for real-time applications, such as chat applications, online games, and streaming services.

161. What is the advantage of using Node.js?
Node.js is fast and scalable. Node.js is easy to learn and use. Node.js is well-suited for real-time applications, such as chat applications, online games, and streaming services. This is because Node.js can handle a large number of connections and can perform non-blocking I/O operations, which makes it ideal for real-time communication.

162. Does Node run on Windows?
Yes, Node.js runs on Windows. Node.js is a cross-platform runtime environment, which means that it can run on a variety of operating systems, including Windows, macOS, and Linux.
Here's How to Land a Top Software Developer Job
Full Stack Development-MEANEXPLORE PROGRAMHere's How to Land a Top Software Developer Job.

163. Can you access DOM in Node?
No, you cannot access the DOM in Node.js. The DOM is a browser-specific API that allows for the manipulation of HTML and XML documents. Since Node.js does not run in a browser, it does not have access to the DOM.

164. Why is Node.JS quickly gaining attention from JAVA programmers?
Node.js is quickly gaining attention from Java programmers because it is fast, scalable, and efficient. Java is a popular server-side technology, but it can be slow and resource-intensive. Node.js, on the other hand, is built on the V8 JavaScript engine, which is known for its speed and performance.

165. What are the Challenges with Node.js?
Node.js is single-threaded, which means that it can only execute one task at a time. Node.js is relatively new compared to other server-side technologies, such as Java and PHP. This means that there needs to be more support and more resources available for Node.js. Node.js is only suitable for applications that require a little memory.

167. What is "non-blocking" in node.js?
In Node.js, non-blocking refers to the ability of the runtime environment to execute multiple tasks simultaneously without waiting for the completion of one task before starting the next. This is achieved through the use of asynchronous I/O operations, which allow Node.js to handle multiple requests concurrently.

168. How does Node.js overcome the problem of blocking I/O operations?
Node.js uses an event-driven, non-blocking I/O model that allows it to handle I/O operations more efficiently. By using callbacks, Node.js can continue processing other tasks while waiting for I/O operations to complete. This means that Node.js can handle multiple requests simultaneously without causing any delays. Additionally, Node.js uses a single-threaded event loop architecture, which allows it to handle a high volume of requests without any issues.

169. How can we use async await in node.js?
To use async/await in Node.js, you'll need to use functions that return promises. You can then use the async keyword to mark a function as asynchronous and the await keyword to wait for a promise to resolve before continuing with the rest of the code.

170. Why should you separate the Express app and server?
Firstly, separating your app and server can make it easier to test your code. By separating the two, you can test your app logic independently of the server, which can make it easier to identify and fix bugs.
Secondly, separating your app and server can make it easier to scale your application. By separating the two, you can run multiple instances of your app on different servers, which can help to distribute the load and improve performance.
Finally, separating your app and server can make it easier to switch to a different server if necessary. By keeping your app logic separate from your server logic, you can switch to a different server without having to make any major changes to your code.

171. Explain the concept of stub in Node.js.
In Node.js, a stub is a function that serves as a placeholder for a more complex function. Stubs are typically used in unit testing to replace a real function with a simplified version that returns a predetermined value. By using a stub, you can ensure that your unit tests are predictable and consistent.

Cleancode

172. What is the framework that is used majorly in Node.js today?
There are many frameworks available for Node.js, but the two most popular ones are Express and Koa.

173. What are the security implementations that are present in Node.js?
One of the most important security features in Node.js is the ability to run code in a restricted environment. This is achieved through the use of a sandboxed environment, which can help to prevent malicious code from accessing sensitive data or causing any damage to the system.
Another important security feature in Node.js is the ability to use TLS/SSL to encrypt data in transit. This can help to prevent eavesdropping and ensure that sensitive data is protected.

174. What is Libuv?
Libuv is a critical component of Node.js, and it's what makes it possible to handle I/O operations in a non-blocking and efficient manner.

175. What are global objects in Node.js?
Global objects in Node.js are objects that are available in all modules without the need for an explicit require statement. Some of the most commonly used global objects in Node.js include process, console, and buffer.

176. Why is assert used in Node.js?
An assert module is an important tool for writing effective tests in Node.js.

167. Why is ExpressJS used?
Express is a great choice for building web applications in Node.js, and its popularity and active community make it a safe and reliable choice for developers of all levels.

168. What is the use of the connect module in Node.js?
Boost Your Coding Skills. Nail Your Next Interview
Full Stack Development-MEANEXPLORE PROGRAMBoost Your Coding Skills. Nail Your Next Interview
The Connect module can be used to handle different types of middleware, such as error-handling middleware, cookie-parsing middleware, and session middleware. Error-handling middleware is used to handle errors that occur during the request/response cycle. Cookie parsing middleware is used to parse cookies from the request header. Session middleware is used to manage user sessions.

169. What's the difference between 'front-end' and 'back-end' development?
Front-end developers focus on the client side of the application, while back-end developers focus on the server side of the application. Both roles are important for building a successful web application and require different skill sets and expertise.

170. What are LTS releases of Node.js?
LTS stands for Long-term support. LTS releases of Node.js are versions that are supported for an extended period, usually for 30 months from the time of release. These releases are typically more stable and reliable than non-LTS releases and are recommended for production use.

Cleancode

171. What do you understand about ESLint?
ESLint is a popular open-source tool that is used to analyze and flag errors and potential problems in JavaScript code.

172. Define the concept of the test pyramid. Please explain the process of implementing them in terms of HTTP APIs.
The test pyramid is a concept that is often used in software testing to illustrate the ideal distribution of different types of tests. The pyramid consists of three layers: unit tests, integration tests, and end-to-end tests. The idea is that the majority of tests should be at the unit level, with fewer tests at the integration and end-to-end levels.
To implement the test pyramid in terms of HTTP APIs, you can start by writing unit tests for each endpoint in the API. These tests should focus on testing the functionality of the endpoint in isolation without making any external requests or dependencies. Once the unit tests are passed, you can write integration tests that test the interaction between different endpoints and components in the API. Finally, you can write end-to-end tests that test the entire API, from the user interface to the database.

173. How does Node.js handle the child threads?
Node.js handles child threads by creating separate instances of the Node.js runtime environment that can be used to execute code in parallel with the main process.
Basics to Advanced - Learn It All!
Caltech PGP Full Stack DevelopmentEXPLORE PROGRAMBasics to Advanced - Learn It All!

174. What is an Event Emitter in Node.js?
An Event Emitter is a Node.js module that facilitates communication between objects in a Node.js application. It is an instance of the EventEmitter class, which provides a set of methods to listen for and emit events. In Node.js, events are a core part of the platform, and they are used to handle asynchronous operations.

175. How to Enhance Node.js Performance through Clustering?
Clustering can be used to improve the performance of HTTP servers, database connections, and other I/O operations. However, it is important to note that clustering does not guarantee a linear increase in performance.

176. What is a thread pool, and which library handles it in Node.js?
A thread pool is a collection of threads that are used to execute tasks in parallel. In Node.js, the thread pool is handled by the libuv library, which is a multi-platform support library that provides asynchronous I/O operations.

177. How are worker threads different from clusters?
Worker threads and clusters are two different approaches to leveraging the power of multiple CPUs in Node.js. While clusters create multiple instances of a Node.js process, each running on a separate CPU core, worker threads provide a way to create multiple threads within a single process.

Cleancode

178. How to measure the duration of async operations?
The console.time and console.timeEnd methods allow you to measure the duration of a block of code. The console.time method is used to start the timer and the console.timeEnd method is used to stop the timer and log the duration to the console.
The performance.now method provides a more precise way to measure the duration of async operations. It returns the current timestamp in milliseconds, which can be used to calculate the duration of a task.

179. How to measure the performance of async operations?
There are several tools and techniques you can use to measure performance, including using the built-in --prof flag, using the perf tool, and using third-party libraries like benchmark.js.

180. What are the types of streams available in Node.js?
There are four types of streams available in Node.js, including readable streams, writable streams, duplex streams, and transform streams.