**1. What is JavaScript?**

JavaScript is a platform-independent,event-driven, interpreted client-side scripting language developed by Netscape Communications Corp. and Sun Microsystems.

JavaScript is a general-purpose programming language designed to let programmers of all skill levels control the behavior of software objects. The language is used most widely today in Web browsers whose software objects tend to represent a variety of HTML elements in a document and the document itself.

But the language is used with other kinds of objects in other environments. For example, Adobe Acrobat Forms uses JavaScript as its underlying scripting language to glue together objects that are unique to the forms generated by Adobe Acrobat.

Therefore, it is important to distinguish JavaScript, the language, from the objects it can communicate with in any particular environment.

When used for Web documents, the scripts go directly inside the HTML documents and are downloaded to the browser with the rest of the HTML tags and content.

**2. How is JavaScript different from Java?**

Don't be fooled by the term Java in both. Both are quite different technologies.

JavaScript was developed by Brendan Eich of Netscape; Java was developed at Sun Microsystems. While the two languages share some common syntax, they were developed independently of each other and for different audiences. Java is a full-fledged programming language tailored for network computing; it includes hundreds of its own objects, including objects for creating user interfaces that appear in Java applets (in Web browsers) or standalone Java applications. In contrast, JavaScript relies on whatever environment it's operating in for the user interface, such as a Web document's form elements.

JavaScript was initially called LiveScript at Netscape while it was under development. A licensing deal between Netscape and Sun at the last minute let Netscape plug the "Java" name into the name of its scripting language. Programmers use entirely different tools for Java and JavaScript. It is also not uncommon for a programmer of one language to be ignorant of the other. The two languages don't rely on each other and are intended for different purposes. In some ways, the "Java" name on JavaScript has confused the world's understanding of the differences between the two. On the other hand, JavaScript is much easier to learn than Java and can offer a gentle introduction for newcomers who want to graduate to Java and the kinds of applications you can develop with it.

**3. What is the official JavaScript website?**

This is a trick question used by interviewers to evaluate the candidate’s knowledge of JavaScript. Most people will simply say javascript.com is the official website.

The truth is- there is no official website for Javascript you can refer to. It was developed by Brendan Eich for Netscape. It was based on the ECMAScript language standard; ECMA-262 being the official JavaScript standard.

**4. What’s relationship between JavaScript and ECMAScript?**

ECMAScript is yet another name for JavaScript (other names include LiveScript). The current JavaScript that you see supported in browsers is ECMAScript revision 3.

**5. What are the various datatypes in javascript?**

Number  
String  
Boolean  
Function  
Object  
Null  
Undefined

**6. What boolean operators does JavaScript support?**

&&, || and !

**7. What is negative infinity?**

It’s a number in JavaScript, derived by dividing negative number by zero.

**8. Is it possible to check if a variable is an object?**

Yes, it is possible to do so. The following piece of code will help achieve the same.

if(abc && typeof abc === "object") {  
console.log('abc is an object and does not return null value');  
}

**9. Can you explain what isNaN function does?**

isNaN function will check an argument and return TRUE (1) if the argument does not seem to be a number.

**10. How do you convert numbers between different bases in JavaScript?**

Use the parseInt() function, that takes a string as the first parameter, and the base as a second parameter. So to convert hexadecimal 3F to decimal, use parseInt ("3F", 16);

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

undefined means a variable has been declared but has not yet been assigned a value. On the other hand, null is an assignment value. It can be assigned to a variable as a representation of no value.

Also, undefined and null are two distinct types: undefined is a type itself (undefined) while null is an object.

Unassigned variables are initialized by JavaScript with a default value of undefined. JavaScript never sets a value to null. That must be done programmatically.

**12. What is the difference between “==” and “===”?**

While “==” checks only for equality, “===” checks for equality as well as the type.

**13. Differentiate between “var a=2” and “a =2”**

The major difference between the two is that one variable is local and the other is global. “var” basically defines the scope of the variable.

When we add var to a variable value assignment, javascript ensures that the variable is confined to whichever function it is assigned to and does not collide with the same name variable within another function.

When we don’t use var, then it is declared as a global function and chances of collision can happen. So it’s always advisable to use “var” before variable value assignment. If needed use an anonymous function for closure.

**14.  What is Javascript namespacing? How and where is it used?**

Using global variables in Javascript is evil and a bad practice. That being said, namespacing is used to bundle up all your functionality using a unique name. In JavaScript, a namespace is really just an object that you’ve attached all further methods, properties and objects. It promotes modularity and code reuse in the application.

**15. What does "1"+2+4 evaluate to?**

Since 1 is a string, everything is a string, so the result is 124.

**16. How about 2+5+"8"?**

Since 2 and 5 are integers, this is number arithmetic, since 8 is a string, it’s concatenation, so 78 is the result.

**17. How to create arrays in JavaScript?**

We can declare an array like this   
var scripts = new Array();   
We can add elements to this array like this

scripts[0] = "PHP";  
scripts[1] = "ASP";  
scripts[2] = "JavaScript";  
scripts[3] = "HTML";

Now our array scrips has 4 elements inside it and we can print or access them by using their index number. Note that index number starts from 0. To get the third element of the array we have to use the index number 2 . Here is the way to get the third element of an array.   
document.write(scripts[2]);   
We also can create an array like this   
var no\_array = new Array(21, 22, 23, 24, 25);

**18. How do you create a new object in JavaScript?**

var obj = new Object(); or var obj = {};

**19. How do you assign object properties?**

obj["age"] = 17 or obj.age = 17

**20. What’s a way to append a value to an array?**

arr[arr.length] = value;

**21. What is this keyword?**

It refers to the current object.

**22. How many looping structures can you find in javascript?**

If you are a programmer, you know the use of loops. It is used to run a piece of code multiple times according to some particular condition. Javascript being a popular scripting language supports the following loops

for  
while  
do-while loop

**23. Are javascript and jQuery different?**

jQuery is a quick as well as concise JavaScript Library that simplifies HTML document traversing, animating, event handling, & Ajax interactions for the purpose of quick web development needs. So although they are not entirely different, both are not the same either!

**24. Explain the strict mode in Javascript.**

The strict mode ensures that if functions are not properly thought it, those are disabled. It also kept a check on potentially unsafe actions and throw errors when it happens.

**25. Is it possible for you to write a one line JavaScript code that concatenates all strings passed into a function?**

The following function should help in producing the desired result

function concatenate()   
{  
  return String.prototype.concat.apply('', arguments);  
}

**26. Explain Javascript closures.**

A basic overview of javascript closures is that it is a stack-frame which is not de-allocated when the function returns.

**27. What is event bubbling?**

Event bubbling describes the behavior of events in child and parent nodes in the Document Object Model (DOM); that is, all child node events are automatically passed to its parent nodes. The benefit of this method is speed, because the code only needs to traverse the DOM tree once. This is useful when you want to place more than one event listener on a DOM element since you can put just one listener on all of the elements, thus code simplicity and reduction. One application of this is the creation of one event listener on a page’s body element to respond to any click event that occurs within the page’s body.

**28. Difference between window.onload and onDocumentReady?**

The onload event does not fire until every last piece of the page is loaded, this includes css and images, which means there’s a huge delay before any code is executed.

That isnt what we want. We just want to wait until the DOM is loaded and is able to be manipulated. onDocumentReady allows the programmer to do that.

**29. How do you change the style/class on any element?**

document.getElementById(“myText”).style.fontSize = “20″;  
-or-  
document.getElementById(“myText”).className = “anyclass”;

**30. How is form submission possible via javascript?**

We can achieve the desired form submission by using the function document.forms[0].submit().

It must be noted that the 0 in the piece of code given above refers to the form index. Say we have multiple forms on a particular page. To make all the form procession unique, we give each form index numbers. The first form will have the index number as 0. The second form will have an incremented number, 1. The third will have 2 and so on.

**31. How JavaScript timers work? What is a drawback of JavaScript timers?**

Timers allow you to execute code at a set time or repeatedly using an interval. This is accomplished with the setTimeout, setInterval, and clearInterval functions. The setTimeout(function, delay) function initiates a timer that calls a specific function after the delay; it returns an id value that can be used to access it later. The setInterval(function, delay) function is similar to the setTimeout function except that it executes repeatedly on the delay and only stops when cancelled. The clearInterval(id) function is used to stop a timer. Timers can be tricky to use since they operate within a single thread, thus events queue up waiting to execute.

**32. How to get CheckBox status whether it is checked or not?**

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

if it will be checked you will get true else false.

**33. How to get value from a textbox?**

alert(document.getElementById('txtbox1').value);

**34. How to get value from dropdown (select) control?**

alert(document.getElementById('dropdown1').value);

**35. How to get value from RadioButtonList control?**

Here id is the name property of the RadioButtonList

function GetRadioButtonValue(id)

        {

            var radio = document.getElementsByName(id);

            for (var ii = 0; ii < radio.length; ii++)

            {

                if (radio[ii].checked)

                    alert(radio[ii].value);

            }

        }

**36. How to detect the operating system on the client machine?**

In order to detect the operating system on the client machine, the navigator.appVersion   
string (property) should be used.