**1. What is JavaScript?**

JavaScript is a programming language that allows you to implement complex things on web pages

JavaScript is a scripting language that enables you to create dynamically updating content, control multimedia, animate images, and pretty much everything else.

**2. What are the advantages of using JavaScript?**

* **Javascript is executed on the client side**  
  This means that the code is executed on the user's processor instead of the web server thus saving bandwidth and strain on the web server.
* **Javascript is a relatively easy language**  
  The Javascript language is relatively easy to learn and comprises of syntax that is close to English. It uses the DOM model that provides plenty of prewritten functionality to the various objects on pages making it a breeze to develop a script to solve a custom purpose.
* **Javascript is relatively fast to the end user**  
  As the code is executed on the user's computer, results and processing is completed almost instantly depending on the task (tasks in javascript on web pages are usually simple so as to prevent being a memory hog) as it does not need to be processed in the site's web server and sent back to the user consuming local as well as server bandwidth.
* **Extended functionality to web pages**  
  Third party add-ons like Greasemonkey enable Javascript developers to write snippets of Javascript which can execute on desired web pages to extend its functionality. If you use a website and require a certain feature to be included, you can write it yourself and use an add-on like Greasemonkey to implement it on the web page.

**3. What are disadvantages of using JavaScript?**

* **Security Issues**  
  Javascript snippets, once appended onto web pages execute on client servers immediately and therefore can also be used to exploit the user's system. While a certain restriction is set by modern web standards on browsers, malicious code can still be executed complying with the restrictions set.
* **Javascript rendering varies**  
  Different layout engines may render Javascript differently resulting in inconsistency in terms of functionality and interface. While the latest versions of javascript and rendering have been geared towards a universal standard, certain variations still exist. [**Website Usability Consultants all over the world**](http://www.nextprise.com/website-usability-consulting.php) make a living on these differences, but it enrages thousands of developers on a daily basis.

### 4. How to use external JavaScript file?

### Lets create .js file with the name message.js, place the following script tag inside the head tag.

<script type="text/javascript" src="message.js"></script>

**5. How can you create an Object in JavaScript?**

In JavaScript, an object is a standalone entity, with properties and type. Compare it with a cup, for example. A cup is an object, with properties. A cup has a color, a design, weight, a material it is made of, etc. The same way, JavaScript objects can have properties, which define their characteristics.

A JavaScript object has properties associated with it. A property of an object can be explained as a variable that is attached to the object. Object properties are basically the same as ordinary JavaScript variables, except for the attachment to objects. The properties of an object define the characteristics of the object. You access the properties of an object with a simple dot-notation:

**objectName.propertyName**

Like all JavaScript variables, both the object name (which could be a normal variable) and property name are case sensitive. You can define a property by assigning it a value. For example, let's create an object named **myCar** and give it properties named **make, model,** and **year** as follows:

**Var myCar = new Object();**

**myCar.make = ‘ford’;**

**myCar.model = ‘mustang’;**

**myCar.year = 1996;**

Unassigned properties of an object are [undefined](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/undefined) (and not [null](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/null)).

**myCar.color; // undefined**

**6. How can you read properties of an Object in JavaScript?**

You can write and read properties of an object using the dot notation as follows −

// Getting object properties

myCar.make// ==> ford

myCar.year // ==> 1996

// Setting object properties

myCar.make = "ford shelby mustang GT500" // <==

myCar.year = 2000 // <== 2000

**7. How can you create an Array in JavaScript and how to read array elements?**

Create an array:

var fruits = ['Apple', 'Banana'];

console.log(fruits.length);

// 2

**8. How to define a anonymous function? Can you assign a anonymous function to a variable? Can you pass a anonymous function as an argument to another function?**

An anonymous function can be defined in similar way as a normal function but it would not have any name. Yes! An anonymous function can be assigned to a variable.

For Example:

var z = function(x, y) {

if ((y \* (x / 100)) < 1) {

return (x + Math.ceil(y \* (x / 100))); }

else if ((y \* (x / 100)) > 1) {

return (x + Math.round(y \* (x / 100))); }

else {

return 0; }

Yes! An anonymous function can be passed as an argument to another function.

**9. How can you get the type of arguments passed to a function?**

Using **typeof** operator, we can get the type of arguments passed to a function. For example −

function func(x){

console.log(typeof x, arguments.length);

}

func(); //==> "undefined", 0

func(1); //==> "number", 1

func("1", "2", "3"); //==> "string", 3

**10. What is the purpose of 'this' operator in JavaScript?**

JavaScript famous keyword this always refers to the current context.

**11. What is callback?**

A callback function is a function passed into another function as an argument, which is then invoked inside the outer function to complete some kind of routine or action.

Here is an example:

function greeting(name) {

alert('Hello ' + name);

}

function processUserInput(callback) {

var name = prompt('Please enter your name.');

callback(name);

}

processUserInput(greeting);

The above example is a synchronous callback, as it is executed immediately.

**12. What is closure?**

A closure is the combination of a function and the lexical environment within which that function was declared.

function init() {

var name = 'Moana'; // name is a local variable created by init

function displayName() { // displayName() is the inner function, a closure

alert(name); // use variable declared in the parent function

}

displayName();

}

init();

**init()** creates a local variable called name and a function called **displayName().** The **displayName()** function is an inner function that is defined inside **init()** and is only available within the body of the  **init()** function. The **displayName()** function has no local variables of its own. However, because inner functions have access to the variables of outer functions, **displayName()**can access the variable name declared in the parent function, init(). However, the same local variables in **displayName()** will be used if they exist.

**13. Which built-in method calls a function for each element in the array?**

forEach() method calls a function for each element in the array.

**14. Which built-in method returns the string representation of the number's value?**

toString() method returns the string representation of the number's value.

**15. What typeof returns for a null value?**

It returns "object".

**16. How to print a web page using javascript?**

JavaScript helps you to implement this functionality using print function of window object. The JavaScript print function **window.print()** will print the current web page when executed.

### 17. What is the use of window object?

The window object is automatically created by the browser that represents a window of a browser.

It is used to display the popup dialog box such as alert dialog box, confirm dialog box, input dialog box etc.

### 18.What is the use of history object?

The history object of browser can be used to switch to history pages such as back and forward from current page or another page. There are three methods of history object.

1. history.back()
2. history.forward()
3. history.go(number): number may be positive for forward, negative for backward.

### 19.What does the isNaN() function?

The isNan() function returns true if the variable value is not a number.

### 20.Difference between Client side JavaScript and Server side JavaScript?

**Client side JavaScript** comprises the basic language and predefined objects which are relevant to running java script in a browser. The client side JavaScript is embedded directly by in the HTML pages. This script is interpreted by the browser at run time.

**Server side JavaScript** also resembles like client side java script. It has relevant java script which is to run in a server. The server side JavaScript are deployed only after compilation.

**21. How to set the cursor to wait in JavaScript?**

The cursor can be set to wait in JavaScript by using the property "cursor". The following example illustrates the usage:

**<script>**

window.document.body.style.cursor = "wait";

**</script>**

### 22. What are the pop up boxes available in JavaScript?

* Alert Box
* Confirm Box
* Prompt Box

### 23. How to change the background color of HTML document using JavaScript?

**<script** type="text/javascript"**>**

document.body.bgColor="pink";

**</script>**