**Javascript interview questions – Part 1**

**1. What is JavaScript?**

JS is a scripting language mostly used to create dynamic websites and enhanced user interfaces. It is an interpreted object-oriented programming language, modeled after C++. It is developed by Netscape and Sun Microsystems in 95, originally called LiveScript. Beside used on objects such as HTML elements and the document itself, it can be used with other kinds of objects in different environments, such as pdf document (Adobe Acrobate).

JS comes in 3 different flavors:

* **core JS** is the basic JS language include the **operators**, **control structures**, **built-in functions** and **objects** that make JS a programming language
* **client-side JS** → extends the core JS to provide **access to browser** **and web document objects** via the **DOM** (Document Object Model)
* **server-side JS** → also called JS platform, currently the most popular is Node.js that runs on top of the V8 engine. It has asynchronous, event-driven I/O and has plenty of modules available. Before that there was a Netscape server-side technology that embedded

<http://en.wikipedia.org/wiki/JavaScript>

<https://developer.mozilla.org/en-US/docs/Web/JavaScript>

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

ECMAscript is a scripting language standardized by ECMA international European Computer Manufacturers Association). It is widely used in client-side scripting on the web and one of its dialects is Javascript (other ones: actionScript...etc).

ECMAScript is yet another name for JavaScript (other names include LiveScript). The current JavaScript that you see supported in browsers is ECMAScript revision 5.

<http://en.wikipedia.org/wiki/ECMAScript>

**3.** **What is DOM?**

DOM (Document Object Model) is W3C (World wide web consortium) standard. It defines a standard for accessing HTML and XML documents. ***"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a web document."***

It provides a structured representation of the document (a tree) and it defines a way that the structure can be accessed from programs, so they can change the document structure, style and content.

DOM tree: Nodes of every document are organized in a tee structure, whose objects can be manipulated by using methods on them.

The W3C DOM standard is separated into 3 different parts:

- **Core DOM** → standard model for any structured document

- **XML DOM** → standard model for XML documents; defines the **objects** and **properties** of all HTML elements and the **methods** to access them.

- **HTML DOM** → standard model for HTML documents; defines the **objects** and **properties** of all XML elements and the **methods** to access them; or in other words: it is a standard for how to **get, change, add** or **delete** HTML elements.

<http://en.wikipedia.org/wiki/Document_Object_Model>

<https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model>

**4.** **What are JavaScript data types?**

There are 3 primitive data types: **Number**, **String**, **Boolean** and other composite data types: **Object (Array)**, **Function**.

There are 2 other primitive values: **null** → denoting null value (note: typeof null is “object”), **undefined** → whose value is undefined

\* **NaN** → it’s also a number whose value is not a number :)

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Grammar_and_types>

**Examples**:

typeof 5 // “number”

typeof “b” // “string”

typeof [1] // “object”

typeof {“a”:1} // “object”

function blah() {console.log(“blah”);}

typeof blah // “function”

function Tea(name) { this.name = name; }

var mytea = new Tea(“green”);

mtypeof Tea // “function”

typeof mytea // “object”

**Note:** JavaScript uses a fixed number of bits, namely 64 of them, to store a single number value. You can use 64-bit chunks to store larger values.

**5.** **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.

There is also parseFloat() function.

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/parseInt>

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/parseFloat>

**Examples:**

parseInt("3F", 16); // 63

parseInt(18, 16); // 24, number 18 is converted to a string

parseInt("18", 10); // 18

parseFloat(“3.14”); // 3.14

To convert a number or string to a binary number use this:

num.toString(2)

**6.** **What does the isNaN function do?**

Its a built-in function that checks if an expression evaluates to NaN or not. Returns true if the argument is NaN - not a number.

[https://developer.mozilla. imorg/en-US/docs/Web/JavaScript/Reference/Global\_Objects/isNaN](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/isNaN)

**Examples:**

isNaN(undefined) // true

isNaN(NaN) // true

isNaN(5) // false

**7**. **What is negative infinity?**

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

Infinity is a property of *global object*, i.e. it is a variable in *global scope*.

isNaN(undefined) // “number”

**8.** **What boolean and comparison, relational operators does JavaScript support?**

Logical (boolean) operators supported by JS are: **&&** (and), **||** (or) and **!** (not). They are used to determine the logic between variables and values. They perform boolean logical comparisons and operate **only on boolean values**.

Comparison (relational) operators are: **==** (is equal to), **===** (is exactly equal to – value and type equal), **!=** (is not equal), **!==** (is not exactly equal – neither value nor type), **>** (is greater than), **<** (is less than), **>=** (is greater than or equal to), **<=** (is less than or equal to). They are used in logical statements to determine equality or difference between variables or values. They **operate on all kinds of data** and they always yield boolean result → both logical and comparison operators evaluate to true or false.

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Logical_Operators>

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Comparison_Operators>

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

Since “1” is a string at the beginning, everything converts to a string, so the result is “124”.

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

Since 2 and 5 are integers, this is number arithmetic – first the addition is performed, and then since “8” is a string, concatenation is performed, and it results in a string “78”.

**11.** **What looping structures are there in JavaScript?**

**- for loop**

for (initializer;conditional expression;loop expression) {//JS statements}

for (key in obj) { //JS statements }

- **forEach**

Array.prototype.forEach(callback, [args])

**- while loop**

while(condition) { //JS statements }

**- do/while loop**

do {//JS statements} while(conditional expression)

<http://www.codecademy.com/glossary/javascript/loops>

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/forEach>

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

There are two ways of creating (an empty) object:

var obj = new Object(); // using constructor

var obj = {}; // object literal

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Working_with_Objects>

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

There are two ways.

obj["age"] = 17

obj.age = 17 // this is only possible if keys are strings

- **Why is one better than the other?**

Dot notation (the second one) is more readable and is usually preferred in simple situations. But bracket notation is more powerful. For example, you might not know the name of the property up front. With dot notation you'd have to know what property you want to access when you're writing the code. Bracket notation is more flexible since you can use a variable to reference the attribute.

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Working_with_Objects>

**Examples:**

var myobj = {“key”:”value”}

var mykey = “key”

myobj[mykey]; // “value”

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

var arr = [1,2,3];

arr.push(4); //[1,2,3,4]

arr[arr.length] = value; //[1,2,3,4,value]

arr.push(7,8); //[1,2,3,4,value,7,8]

**15**. **How do you submit a form using JS?**

Check the answer here:

<http://www.javascript-coder.com/javascript-form/javascript-form-submit.phtml>

**16.**  **How to create arrays in JS?**

There are a couple of ways:

- create an array literal by putting its values directly into square brackets.

var anArray = [1,2,”red”];

- with Array() constructor function:

var a = new Array();

- add elements to an array: a[0]=5;a[1]=true;

\*note: if you pass only a single number to the constructor → it specifies the length of an array

- you can also pass array elements to the Array() constructor

var myArray = new Array(1,2,3);

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array>

Array uses keys (indices) to access its elements. A key is a general term used to describe a piece of information used to look up a piece of data. An index is just a certain kind of key, a numeric key.

**17. How to read and write a file using JS?**

This can be done with server-side JS, using Node.js

- further reference: <http://nodejs.org/api/fs.html>

- using HTML5 File API <http://www.html5rocks.com/en/tutorials/file/dndfiles/>

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

By displaying one of the properties of the navigator object, which is a property of the global window object: **navigator.appVersion**

<https://developer.mozilla.org/en-US/docs/Web/API/Window.navigator>

**19. How do you target a specific frame from a hyperlink in JS?**

<http://javascript.about.com/library/bltarget.htm>

**20. What are fixed-width table and its advantages in JS?**

When you use the CSS style ‘table-layout: fixed’ to specify a fixed width table, browser renders the table based on the widths of the columns in the first row. It results in a faster display of the table, which is especially beneficial for large tables.

Otherwise, the browser has to wait until all data is downloaded and then figure out the best width for each of its columns. It can be pretty slow for large tables.

**21.** **Example of using Regular Expressions for syntax checking in JS?**

Regex can be used when validating data from the input form.

Simple example - match a phone number:

[“123-456”].match(/^[1-9()-]+$/)

**22. Where are cookies actually stored on the hard disk?**

Depends on the user's browser and OS. There are 2 different kind of cookies:

1. Session cookies - temporary, stored within a particular browser session. They are erased when you close a browser session.
2. Persistent cookies - stayed in one of the browser’s subfolders until you delete them manually, or until they ‘expire’. Duration period is contained within persistent cookies file.

<http://www.allaboutcookies.org/cookies/cookies-the-same.html>

**23. What can JS programs do?**

They make web-pages interactive by generating HTML documents on the fly without accessing the Web server. The user can be given control over the browser, like User input validation, then simple computations can be performed on user's machine. The user's browser, OS can be detected, also Date and Time can be handled.

**24.** **How to set a HTML document background color?**

document.body.style.backgroundColor = “red”;

**25. How is Javascript different from Java?**

- Javascript is developed by Brendan Eich at Netscape, while Java was developed at Sun Microsystems

- JS is a OOP scripting languages, Java is a OOP programming language

- JS provides interactivity to webpages

- Java creates applications that run in a virtual machine (This is important to understand about Java. To provide some context, C/C++ gets compiled directly into machine code which runs directly on the processor (making the binaries machine dependent). Java on the other hand also gets compiled but to something called byte code (different from machine code). That byte code is machine **independent** so you can run a Java program on any machine that has a Java runtime. Java applications are run in a virtual machine which does some fast (and very well optimized) translation during runtime.) or browser while JS code is run in a browser only (now days there is server-side JS too...)

- Java code needs to be compiled, while JS code is all in text (interpreted vs compiled)

- they require different plug-ins

- the 2 languages don’t rely on each other, and are intended for different purposes

1) Java is statically typed and Javascript is dynamically typed.

2) Java is a compile to VM language while Javascript is a scripting language that gets run in an interpreter.

3) Both could be technically considered OOP languages but Java uses class inheritance while Javascript uses prototypal inheritance.

4) Java is primarily used on the back end while Javascript is commonly used in both.

5) All modern browsers support Javascript whereas Java is an optional plugin.

“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.”

**26. How can JS make a website easier to use? That is, are there certain JS techniques that make it easier for people to use a Web site?**

Some good examples:

1) Using javascript widgets so that small parts of the page can be interactive instead of fetching and reloading a whole new page after each interaction.

2) Using AJAX to accomplish the same ends as #1

3) Validation - so the user doesn't have to wait until they've submitted a form to know whether or not their input is valid

4) Auto focus on input elements - nice usability enhancement that can often save clicks

5) Tooltips (always helpful)

**27. How can JS be used to improve the ‘look and feel’ of a Website? By the same token, how can JS be used to improve the user interface?**

- Javascript used to be used a lot for animations but CSS3 animations are really powerful and more performant (since they're run on the GPU instead of the CPU). Still there are certain things you can't easily do with CSS3 so javascript is still often used.

- The bootstrap JS page is probably a good place to pull inspiration from. These are common components that can't be implemented in CSS alone

<http://getbootstrap.com/javascript>.

- Parallax effects can only be accomplished with Javascript

**28. Are you concerned that older browsers don't support JS and thus exclude a set of Web users?**

I'd leave coming up with a browser support matrix up to the PM. It depends on the target audience. Adding legacy browser support can significantly slow down development time and/or worsen the product for regular users so it's usually a better tradeoff to exclude ancient browsers. Some stats to keep in mind:

- Less than 1% of browsers by marketshare don't support JS

- Limiting support to ie8+ excludes 7% of the market (mostly large enterprises)

<http://thenextweb.com/insider/2014/02/01/ie11-passes-ie10-market-share-firefox-slips-bit-chrome-gains-back-share/>

**29.** **In a pop-up browser window how do you refer to the main browser that opened it?**

- you use window.opened to refer to the main window from pop-up windows

**30.**  **Methods GET and POST in HTML forms – what's the difference?**

GET method is visible in the browser address bar – parameters passed in the query string - and has limits on the amount of data it can send (around 2kb).

POST method variables are not displayed in the URL – parameters passed in the request body and there is no limit on amount of data (only amount of data in one key/value pair).

**GET**:

- Parameters remain in the browser history, because they are part of the URL

- Can be bookmarked

- On BACK button click GET requests are re-executed

- Parameter data is limited to what we can stuff into the request line (URL). Safest up to 2kb

- Easier to hack

- Less secure compared to POST → its saved in the browser history and server logs in plaintext

- GET method should not be used when sending passwords, and other sensitive information

- can be cached

- only ASCII characters allowed

- 7607 character max size

**POST**:

- parameters not saved in browser history

- cannot be bookmarked

- on BACK button click the browser alerts the user that data will be resubmitted

- can send parameters, including uploading files to the server

- more difficult to hack

- no restriction on data type, binary data allowed

- no restriction on data length

- POST method used when sending passwords and other sensitive data

- POST method variables are not displayed in the URL

- NOT cached

- 8MB maximum size

**31. How to write a script for a “select” list using Javascript?**

document.formName.selectName.options[0]

document.formName.selectName.options[1]

document.formName.selectName.options[2]

Properties: **text**, **value**, **length**

document.formName.selectName.options[0].text;

document.formName.selectName.options[0].value;

Set options item to null:

document.formName.selectName.options[0] = null;

**32.** **What does “Access denied” IE error mean?**

IE raises this error as a part of its security zone features. It is raised when attempting to access a resource in the “Local Intranet” zone from an origin in the global Internet zone, when using default settings. To get around this error, you can add the origin site to the “Trusted site” security zone.

**33.** **How to embed JS in a webpage?**

<script type=”text/javascript language=”javascript”> //JS statements </script>

→ those two attributes are optional in HTML5

**34. What are the problems associated with using JS and are there JS techniques that you discourage?**

There are a couple of problems, such as

1. Different browsers have their own quirks and when developing you need to account for cross browser compatibility. Using jQuery can help with this.
2. Global namespace - be careful when declaring variables/objects globally (they get attached to the window object), that can lead to conflicts
3. Type conversion is often a source of failures (loose typing, e.g. “==” vs “===”)
4. Memory leaks (MVCs) when keeping a reference to views that no longer exist via event listeners
5. Using closures that preserve the state of the outer environment

**35.** **How to get the contents of an input box using JS?**

document.getElementById("inputElementId").value

**36. How to determine the state of a checkbox using JS?**

The checkbox input element has a value ‘checked’ that evaluates to true or false.

Example:

document.getElementById(“check-box”).checked // ‘true’ or ‘false

**37.** **How to set the focus element using JS?**

document.getElementById("inputElementId").focus()

**38.**  **What is the difference between an alert box and confirmation box?**

An alert box displays only one button which is OK button, whereas Confirmation box displays two buttons OK and Cancel.

**39. What is a prompt box?**

It prompts user for an input.

**40. Can JS code be broken in different lines?**

Yes. You can split statements into as many lines as you want. For example:

var

myvar = 2 +

3;

is totally valid. The interpreter will keep reading until the semicolon. That said, using a consistent coding convention is very important for readability.

**41. Difference between normal and associative array**

'Normal' array → uses integer numbers as indexes. And it is ordered.

Associative array = Object → maps key to value (pairs). Keys are strings. Unordered.

**42. How would you compare if 2 objects are the same?**

You can loop through their properties or use Object.toJSON(obj)

<http://stackoverflow.com/questions/1068834/object-comparison-in-javascript>