# What are the features of JavaScript?

Following are the **features** of JavaScript:

* It is a **lightweight, interpreted** programming language.
* It is designed for creating **network-centric** applications.
* It is complementary to and **integrated** with Java.
* It is an **open** and **cross-platform** scripting language.

# What are JavaScript Data Types?

The **data types** supported by JavaScript are:

* Undefined
* Null
* Boolean
* String
* Symbol
* Number
* Object

# What is the use of isNaN function?

isNan function returns true if the argument is not a number otherwise it is false.

# **What is negative infinity?**

Negative Infinity is a number in JavaScript which can be derived by dividing negative number by zero.

# **What are undeclared and undefined variables?**

Undeclared variables are those that do not exist in a program and are not declared. If the program tries to read the value of an undeclared variable, then a runtime error is encountered.

Undefined variables are those that are declared in the program but have not been given any value. If the program tries to read the value of an undefined variable, an undefined value is returned.

# **What are global variables? How are these variable declared and what are the problems associated with using them?**

Global variables are those that are available throughout the length of the code, that is, these have no scope. The var keyword is used to declare a local variable or object. If the var keyword is omitted, a global variable is declared.

Example:

// Declare a global globalVariable = "Test";

The problems that are faced by using global variables are the clash of variable names of local and global scope. Also, it is difficult to debug and test the code that relies on global variables.

# **Is JavaScript a case-sensitive language?**

Yes, JavaScript is a**case sensitive** language.  The language keywords, variables, function names, and any other identifiers must always be typed with a consistent capitalization of letters.

# **What is the use of ‘debugger’ keyword in JavaScript code?**

Using the ‘debugger’ keyword in the code is like using breakpoints in the debugger.

To test the code, the debugger must be enabled for the browser. If debugging is disabled for the browser, the code will not work. During debugging the code below should stop executing, before it goes to the next line.

# **What is JavaScript Hoisting?**

Using ‘JavaScript Hoisting’ method, when an interpreter runs the code, all the variables are hoisted to the top of the original /current scope. If you have a variable declared anywhere inside the JavaScript code then it is brought to the top.

This method is only applicable for the declaration of a variable and is not applicable for initialization of a variable. Functions are also hoisted to the top, whereas function explanations are not hoisted to the top.

Basically, where we declared the variable inside the code doesn’t matter much.

# **What is JavaScript ‘Strict Mode’?**

‘Strict mode’ is a restricted variant of JavaScript.

Usually, JavaScript is ‘not very strict’ in throwing errors.

But in ‘Strict mode’ it will throw all types of errors, even the silent errors. Thus, the process of debugging becomes easier. And the chances for making mistake for the developer is reduced.

# What are the characteristics of JavaScript ‘Strict Mode’?

Given below are the characteristics of JavaScript ‘Strict Mode’:

* ‘Strict Mode’ will stop developers from creating global variables.
* Developers are restricted from using duplicate parameters.
* Strict mode will restrict you from using JavaScript keyword as a variable name or function name.
* Strict mode is declared with ‘use strict’ keyword at the beginning of the script.
* All browser support strict mode.

# **What are Self Invoking Functions?**

They are also known as ‘Immediately Invoked Function Expressions’ or ‘Self Executing Anonymous Functions’. These functions are invoked automatically in the code, hence they are named as ‘Self Invoking Functions’.

Usually, we define a function and invoke it, but if we want to execute a function automatically where it is explained, and if we are not going to call it again, we can use anonymous functions. And these types of function have no name.

**The syntax for the Self-Invoking function:**

(function () {

return () } () ;

# What is the difference between ‘var’ and ‘let’ keyword?

Answer: The Differences are as follows:

| Var | let |
| --- | --- |
| ’var’ keyword was introduced in JavaScript code from the beginning Stage itself. | ‘let’ keyword is introduced in 2015 only. |
| ’Var’ keyword has function scope. The variable defined with var is available anywhere within the function | A variable declared with ‘let’ keyword has a scope only with in that block. So, let has a Block Scope. |
| The variable declared with ‘var’ be hoisted | The variable declared with ‘let’ be hoisted |

# What is the difference between ‘==’ and ‘===’?

Both ‘==’ and ‘===’ are comparison operators.

| **‘==’ operator** | **‘===’ operator** |
| --- | --- |
| It is known as ‘Type Converting Operator’ | It is known as ‘Strict Equality Operator’ |
| It compares Value, do not compare type | It compares both value and type. |

# What is the difference between ‘let’ and ‘const’?

| let | const |
| --- | --- |
| using ‘let’ we can change the value of variable any number of times | using ‘const’, after the first assignment of the value we cannot redefine the value again |
| Consider the code { let first\_num =1; first\_num=2; document. write (first\_num); } Here the code will give an output, since the change in value of first\_num is possible. | Consider the code { const second\_num =1; second\_num=2; document. write (second\_num); } Here the code will produce an error, since the ‘second\_num’ is assigned with a second value. |

# What is the difference between ‘null’ and ‘undefined’?

Both the keywords represent empty values**.**

**The differences are:**

In ‘undefined’, we will define a variable, but we won’t assign a value to that variable. On the other hand, in ‘null’ we will define a variable and assign the ‘null’ value to the variable.

type of (undefined) and type of (null) object.

# **What is Closure? Give an example.**

**Closures** are created whenever a variable that is defined outside the **current scope** is accessed from within some inner scope. It gives you access to an outer function’s scope from an inner function. In JavaScript, closures are created every time a function is created. To use a closure, simply define a function inside another function and expose it.

# **What is null in JavaScript?**

The value null represents the intentional absence of any object value.

This is one of JavaScript's primitive values.

**For Example,**

Var myvar = null;

console.log(myvar); //This will print null

# **What is a ternary operator?**

The ternary or conditional is an operator that is used to make a quick choice between two options based on a true or false test.

# **What are Anonymous functions?**

Anonymous functions are functions without having any name and won't do anything on their own. These are generally used along with an event handler.

# Difference between “undefine” and “NULL” Keywords?

When you define a var but not assign any value. typeof(undefine)=> undefine  
Null- manually done. typeof(null)=> object

# What is [SetTimeout()](https://www.geeksforgeeks.org/java-script-settimeout-setinterval-method/)?

When you setTimeout it becomes asynchronous and it has to wait on the stack to get everything got finished

# **What are the scopes of a variable in JavaScript?**

The scope of a variable is the **region** of your program in which it is **defined**. JavaScript variable will have only two scopes.

**Global Variables** − A global variable has global scope which means it is visible everywhere in your JavaScript code.

**Local Variables** − A local variable will be visible only within a function where it is defined. Function parameters are always local to that function.

# **What is the purpose of ‘This’ operator in JavaScript?**

The JavaScript **this** keyword refers to the object it belongs to. This has different values depending on where it is used. In a method, this refers to the owner object and in a function, this refers to the global object.

# **What is Callback?**

A **callback** is a plain JavaScript function passed to some method as an argument or option. It is a function that is to be **executed** after another function has finished executing, hence the name ‘**call back**‘. In JavaScript, functions are objects. Because of this, functions can take functions as arguments, and can be returned by other functions.

# **What are the ways to define a variable in JavaScript?**

The three possible ways of defining a variable in JavaScript are:

* **Var** – The JavaScript variables statement is used to declare a variable and, optionally, we can initialize the value of that variable. Example: var a =10; Variable declarations are processed before the execution of the code.
* **Const** – The idea of const functions is not allow them to modify the object on which they are called. When a function is declared as const, it can be called on any type of object.
* **Let** – It is a signal that the variable may be reassigned, such as a counter in a loop, or a value swap in an algorithm. It also signals that the variable will be used only in the block it’s defined in.

# **What is the difference between Local storage & Session storage?**

**Local Storage** – The data is not sent back to the server for every HTTP request (HTML, images, JavaScript, CSS, etc) – reducing the amount of traffic between client and server. It will stay until it is manually cleared through settings or program.

**Session Storage** – It is similar to local storage; the only difference is while data stored in local storage has no expiration time, data stored in session storage gets cleared when the page session ends. Session Storage will leave when the browser is closed.

# **What is the difference between the operators ‘==‘ & ‘===‘?**

The main difference between “==” and “===” operator is that formerly compares variable by making **type correction** e.g. if you compare a number with a string with numeric literal, == allows that, but === doesn’t allow that, because it not only checks the value but also type of two variable, if two variables are not of the same type “===” return false, while “==” return true.

# **What is the difference between undeclared & undefined?**

Undeclared variables are those that do not **exist** in a program and are not declared. If the program tries to read the value of an undeclared variable, then a **runtime error** is encountered. Undefined variables are those that are declared in the program but have not been given any value. If the program tries to read the value of an undefined variable, an undefined value is returned.

# **What is the difference between Call & Apply?**

The **call()** method calls a function with a given this value and arguments provided individually.

The **apply()** method calls a function with a given this value, and arguments provided as an array.

# **What do mean by NULL in Javascript?**

The NULL value is used to represent no value or no object. It implies no object or null string, no valid boolean value, no number and no array object.

# **What is the function of delete operator?**

The delete keyword is used to delete the property as well as its value.

Example

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

delete student.age;