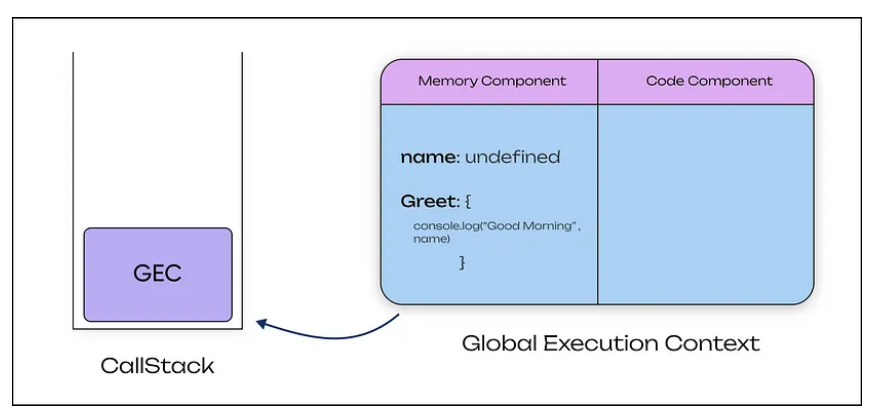
**Variables in JavaScript**

**Topic – 1 : GEC**

There are two different types of execution context which are created in JavaScript by JS engine:

1. **Function Execution Context (FEC)-**This context is created only when JS engine encounters any function call inside code.
2. **Global Execution Context (GEC)-**When the JavaScript code is executed JS engine creates Global Execution context by creating a global object called**window object**in JavaScript. It is the root level context which is created for code in global scope. There can be only one Global Execution Context for every JavaScript file.



GEC / Global Execution Context is also called the base/default execution. Any JavaScript code which does not reside in any function will be present in the global execution context. The reason behind its name 'default execution context' where the code begins its execution when the file first loads in the web browser. GEC performs the two following tasks:

* Firstly, it creates a global object where it is for Node.js and Window object for the browsers.
* Secondly, reference the Windows object to 'this' keyword.
* Create a memory heap in order to store variables and function references.
* Then it stores all the functions declarations in the memory heap area and the variables in the GEC with initial values as 'undefined'.

**Topic – 2 : Variables**

Variables are used to store data in JavaScript. Variables are used to store reusable values. The values of the variables are allocated using the assignment operator(“=”).

**JavaScript Identifiers**

JavaScript variables must have unique names. These names are called Identifiers.

**Basic rules to declare a variable in JavaScript:**

* These are case-sensitive
* Can only begin with a letter, underscore(“\_”) or “$” symbol
* It can contain letters, numbers, underscore, or “$” symbol
* A variable name cannot be a reserved keyword.

**JavaScript** is a dynamically typed language so the type of variables is decided at runtime. Therefore there is no need to explicitly define the type of a variable. We can declare variables in JavaScript in three ways:

* var
* let
* const
* automatic

**Note:**In JavaScript, variables can be declared automatically.

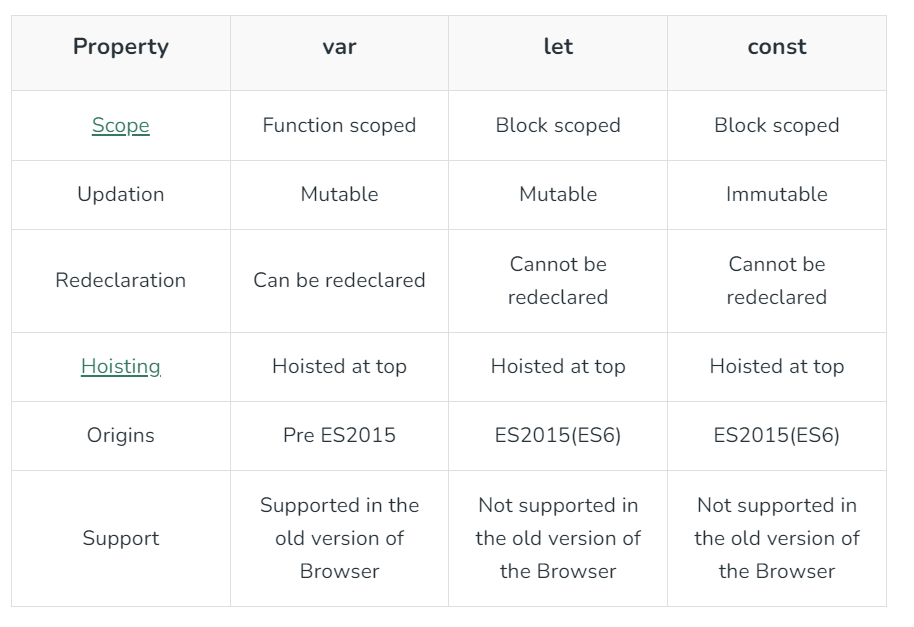
All these keywords do the basic task of declaring a variable but with some differences Initially, all the variables in JavaScript were written using the var keyword but in ES6 the keywords let and const were introduced.

**When to Use var, let, or const**

* We declare variables using const if the value should not be changed
* We use const if the type of the variables should not be changed such as working with Arrays and objects
* We should use let if we want mutable value or we can not use const
* We use var only if we support old browser.

**Syntax :**

// Declaration using var  
var geek = "Hello Geek"  
   
// Declaration using let   
let $ = "Welcome"   
   
// Declaration using const   
const \_example = "Gfg"



**Topic – 3 : Hoisting**

JavaScript Hoisting is the behavior where the interpreter moves function and variable declarations to the top of their respective scope before executing the code. This allows variables to be accessed before declaration, aiding in more flexible coding practices and avoiding “undefined” errors during execution.

**What is Hoisting in JavaScript?**

Hoisting is the default behavior in JavaScript where variable and function declarations are moved to the top of their respective scopes during the compilation phase. This guarantees that regardless of where these declarations appear within a scope, they can be accessed throughout that scope.

Features of Hoisting

* Declarations are hoisted, not initializations.
* Allows calling functions before their declarations.
* All variable and function declarations are processed before any code execution.
* Undeclared variables are implicitly created as global variables when assigned a value.