**Global Object in Node.JS**

In the browser, the global object is called the window.

But in Node.js, the global object is called global.

Moreover, global is a bit different from window global object.

Like,

Variables are scoped to the file where it is get declared or that module. This means that every new file that we create will have its own scope for these variables. So in order to run code of those file by node command, make sure you’re in the correct directory & file like( *node ‘1.global’*

Example:

lethello="Hello World from Node.js";

*console.log*(*global.*hello);

Output:  
undefined

So we can use JavaScript in NodeJs file because Node.js uses Chrome’s V8 Interpreter. Node.js works with primitives, objects, arrays, and functions, just like our browser does.

Ex:

lethello="Hello World from Node.js",

justNode= *hello.slice*(17);

*console.log*(`Who let ${justNode} out?`);

Output:  
Who let Node.js out?

*console.log*(\_\_dirname);

*console.log*(\_\_filename);

These are the common JS module pattern which we’re going to use to import other code into the file.

One way to import other modules by using ***require*** function:

Ex:

const path = require(“path”);

Here, we’re *importing the path module* from Node.

Path module is one of those core modules that’s part of Node.js but in order to use it we’ve to import it for us to be able to use all of these functions.

**Why might I want to use one of these functions?**

Well, I want to pull out just global.js from the file extension that we’re getting from the file name variable.

Ex:

// *path module Imorted*

const *path* = *require*("path");

*console.log*(`The file name is ${*path.basename*(\_\_filename)}`);

Output:  
The file name is 1. global.js

**To find what are those elements (properties or Methods) in global object of Node:**

*for* (letkeyin global) {

*console.log*(key);

}

Output:  
clearInterval  
clearTimeout  
setInterval  
setTimeout  
queueMicrotask  
performance  
clearImmediate  
setImmediate