# Module in Node.js

Node.js uses the Module architecture to simplify the creation of complex applications. The module is like libraries in C, C #, Java ... Each module contains a set of functional functions related to an object of Module.

## Module in Node.js

Node.js uses the Module architecture to simplify the creation of complex applications. The module is like libraries in C, C #, Java . Each module contains a set of functions that are related to an "object" of the Module. For example, http is a Module containing specific functions related to HTTP settings. Node.js provides several additional core modules to assist us with accessing files on the system, creating HTTP, TCP / UDP servers, and other useful small utility functions.

Before using the Module, you simply need to declare with the require () function, as follows:

```
var  http  = require ( "http" );
```

Node.js is just an environment, you have to do everything yourself!

require () is a function that returns a reference to a specific Module. In the case of the above code, we are declaring a reference to the http module and saving it to the http variable.

In the above code, they pass a parameter as the name of the Module. This tells the Node to find a Module named http in the node_modules directory of the application. If it doesn't, Node will continue to find that Module in the global node installation directory.

The command to check the global directory to install node_modules, open the CMD command line interface and type the following command:

```
npm root  - g
```

Back to the problem, you can specify the file by passing the parameter as the relative path ./path/to/my/module.js or absolute /path/to/my/module.js

```
var  myModule  = require ( './myModule.js' )
```

In short, Module is code that is packaged together. The code in a Module is usually private - meaning functions, variables defined and accessed by the inside of the Module. But, you can point out the api are functions and / or variables to use outside the Module. By using an export object, see the following example:

```
var  PI  = Math . PI ;  exports . dientich  = function ( r ) { return  PI  *  r
```

The code above creates a PI variable and it is only accessible in the Module we are defining. By using exports to output two functions used outside the Module are dientich () and chuvi (). So, suppose we are writing code on the ./myModule.js file, the reference variable myModule can call dientich () and chuvi () functions.

# Global Scope in Node.js

Node.js is a server-enabled environment that runs JavaScript on the server and runs on Google's V8 JavaScript engine. Thus, we should implement the code like when using client-side programming. For example, we should limit the use of global variables. However, if you want to use it, you can easily create a global variable by defining the variable name without the var keyword, as follows:

```
globalVariable  = 1 ;  globalFunction  = function () { . };
```

Again, global variables should be limited to the maximum. Therefore, be careful and remember to use the var keyword to declare the variable.

**According to Tutorialspoint**

Previous article: Instructions for installing Node.js

Next lesson: Hello World program in Node.js