Module export & require — Lec-4

* How To RUN two JS Files togenter.

app.JS
xyz.JS

```
app JS

require ("./xyz.JS")

Var a = 10;
```

↑ The Code within xyz.JS will Run First ie it will be executed line by line First then code within app.JS will RUN

* whenever You create a Saparate Module & You require that Module This Code will RUN but You Cannot access the Variables, Methods, functions of one Module into another Simply by Requiring

Note: Modules are Protected By Default

Module Protects their variables & Functions from leaking.

* To Access the functions & Variables from a module, you need to export them from the Module and import that functions in Another File.

To export we use →

Module. exports = function_name ;

To import we use →

const function_name = require('File_name')

To export more than one Function or variable

Module. exports = { }
↑
wrap them in brackets

Points to Remember →

From a module you cannot access its variable & Functions unless a Module wants them to be accessed or exports them.

If module. exports was not there and we could access the variable and Methods by requiring ~~them~~ the Whole Module, it could lead to Redundancy & inconsistencies

require ('./app.js')
or
require ('./app')

} Both works

The pattern we have seen till now was Common JS Module (CJS)

The other modules used are ES Modules (MJS)

By Default CJS is used, to enable MJS write "type": "module" in Package.JSON File. After this we Cannot use CJS Method ie Require statement.

Common JS Module	ES Module
→ module.exports require ()	→ import export
→ By Default used by Node	→ By default used by React, Angular
→ 'type': 'commonjs'	→ 'type': 'module'
→ older way	→ Newer way
→ Synchronous & Runs in non Strict mode	→ Asynchronous & Runs in Strict mode.

Note →

Module. exports is an empty object in JS File

Console. log (module. exports)

We can attach properties to it in two ways

Module. exports = { x, fun-name }

OR

module. exports. x = x ;
module. exports. Fun_name = Fun_name;

→ Require Multiple functions from a same File which were defined in Different Files →

Make a new File →

Const { function-from-File1 } = require ('File1')
Const { function-from-File2 } = require ('File2')
module. exports { Function-from-File1, Function-from-File2

Lec -5

Diving Deep Into NodeJs Github Repo

What is IIFE

When we do require('./path')

All the code of this module is wrapped inside a function (IIFE)

IIFE → Immediately Invoked Function Expression.

```
(function () {

        // All Code of Module Runs inside this

})()
```

↑ These Brackets are <u>important</u>

Before Execution All the requiries are converted into IFFE or wrapped inside IFFE