Diving Deep Into NodeJS Github Repository.

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 the code of module runs inside this
})()
```

↑ these brackets are imp.

Before Execution All the code in require are converted into IIFE or wrapped inside IIFE.

It keeps variable & functions private and safe. The code inside IIFE will not interfere with the file's code

Q// How are variables & function private in different module?

lets support we have a file

```
function x(){
    const x = 10;
    function b(){
        console.log('b');
    }
}
console.log(a) <- this a is not accesible here.
```

~~inter~~ when ever you create a function it creates its own space and scope

Modules work the same way.

whenever you create a module all the code that you write in a module is wrapped inside a function and then executed.

This is why you cannot access the data of the module without export.

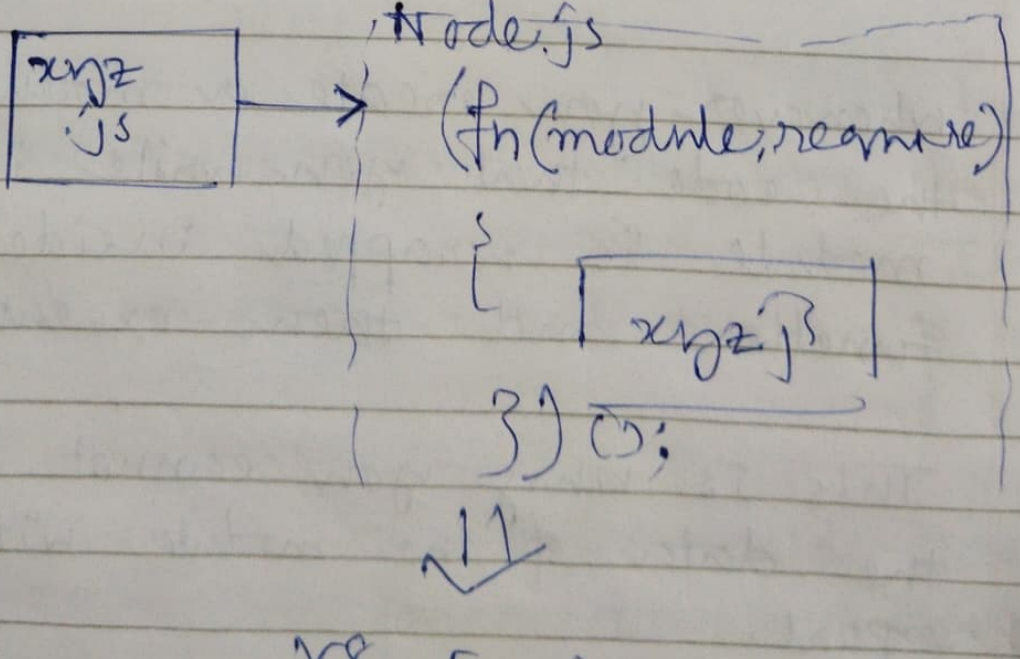
Q// How do you get access to module.exports, where does it come from?

At the end of the day all the JS code is wrapped inside a function & gets executed (IIFE)

```
(function(module, require){
    require('./path');
    module.exports = {}
})();
```

↳ Given by Node.

Node.js passes module as a parameter to IIFE in which the code is wrapped.



Node Js takes your code wraps it inside IIFE & sends it to V8 for execution. V8 Engine knows how to execute IIFE

This is all about IIFE & Its working.

* 5 Steps Mechanism of Reanire('./path')

① Resolving the module

→ ./ local path

→ .json

→ node : module.

It sees from where the data is coming and accordingly it resolve the module.

② Loading the module

File content is loading according to file type.

③ Wraps Inside IIFE

④ Evalution → code is executed & returns module.exports