IIFE: **immediately invoked function expression**

**What**

An immediately-invoked function expression is basically a function with a lexical scope enclosed within the [Grouping Operator](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Grouping) () but you combine it with a function expression () so JavaScript engine directly interprets the function when it’s defined and inner statements are not accessible from outside.

**Why**

**When**

**If two iife are written we must specify the ; to execute them properly**

**Encapsulation achieved by IIFE and reason**

**Revealing Module Pattern**

Modules are most commonly used design patterns in JavaScript. It provides loose-coupling in order to have a well-structured code. Since IIFE provides private scope statements, when we use it with this pattern, we’ll have a well-designed encapsulated objects.

You can see how I designed the code above with this pattern:

const store = (() => {  
 **/\*\*  
 \* private members => only private to constructor function scope  
 \*/**  
 const \_store = ["Apple", "Banana", "Mango", "Orange"]; const \_print = (data) => {  
 console.log("Inventory: ", data.toString());  
 }  **/\*\*  
 \* public members  
 \*/**  
 return {  
 getInventory: () => \_print(\_store),  
 }  
})();store.getInventory();

*Returns* ***Inventory: Apple,Banana,Mango,Orange***

Js is not based on OOP

1. Iife is a function execution which is not hoisted to the global execution.That’s why We can be able to override it.
2. It only calls once and automatically. We can’t call it again once it call.