An immediately invoked function expression (IIFE for short) is a JavaScript design pattern that declares an anonymous function and immediately executes it.

Syntax: IIFEs follow a particular syntax as shown below.

```
(function (){
    // Function  Here.
})();
```

immediately invoked

This type of function is called immediately invoked as these functions are executed as soon as they are mounted to the stack, it requires no explicit call to invoke the function. If we look at the syntax itself we have two pairs of closed parentheses, the first one contains the logic to be executed and the second one is generally what we include when we invoke a function, the second parenthesis is responsible to tell the compiler that the function expression has to be executed immediately.

We use IIFEs :

IIFEs are useful because they can define local variables that aren't accessible outside the IIFE. For example, IIFEs are often used to execute JavaScript in the browser without polluting global

Scope. IIFEs have their own scope i.e. the variables you declare in the Function Expression will not be available outside the function.

```js
(function() {
    var answer = 42;
    console.log(answer)
})();


console.log(answer)
```

```
42                                                      script.js:3
❌ ▶Uncaught ReferenceError: answer is not               script.js:6  🔍
   defined
       at script.js:6:15
```

```js
const cost = 40;
const total = (function() {
  const total_cost = product.salesTaxRate * subtotal;
  return cost + total_cost;
})();
```

1. IIFEs have their own scope i.e. the variables you declare in the Function Expression will not be available outside the function.
2. Similarly to other functions IIFEs can also be named or anonymous, but even if an IIFE does have a name it is impossible to refer/invoke it.
3. IIFEs can also have parameters

```
(function(dt) {
    console.log(dt.toLocaleTimeString());

})(new Date());
```

```
1:45:35 AM
```
>

```
void function() { console.log('Hello, World'); }();
/*we can omit the parenthesis around IIFEs if you use the IIFE with a unary operator,
 like the void operator. */
```

top   Filter   Default levels   2

Hello, World   script.js:1
>