Callback Hell

Tuesday, 22 August 2023 12:48 AM

→ **Callback Hell:** It is a phenomena in which there is a callback inside another callback, which is inside another callback and so on.
thus, our code starts to grow horizontally instead of vertically.
This is very unreadable & unmaintainable. this structure is also called pyramid of doom.

```
1
2  const cart = ["shoes", "pants", "kurta"];
3
4  api.createOrder(cart, function () {
5
6      api.proceedToPayment(function () {
7
8          api.showOrderSummary(
9
10             function () {
11                 api.updateWallet()
12             }
13
14         )
15
16     })
17
18 })
19
20
```

→ In this updateWallet() depends on showOrderSummary() which in turn depends on proceedToPayment() which finally depends on createOrder() fn.
→ So, when showOrderSummary() runs, only then can updateWallet() run.

→ **Inversion of control:** This refers to losing the control of code when using callbacks.

```
1
2  const cart = ["shoes", "pants", "kurta"];
3
4  api.createOrder(cart, function () {
5
6      api.proceedToPayment()
7
8  })
9
```

→ Here, we know that proceedToPayment() is an important piece of code. But we've given the control of running that callback fn to createOrder(). What if createOrder() doesn't run? Then our callback fn would also not run, this is inversion of control.
Thus, this is a problem we face when using callbacks.