



Today's agenda

observer design pattern

UML diag.

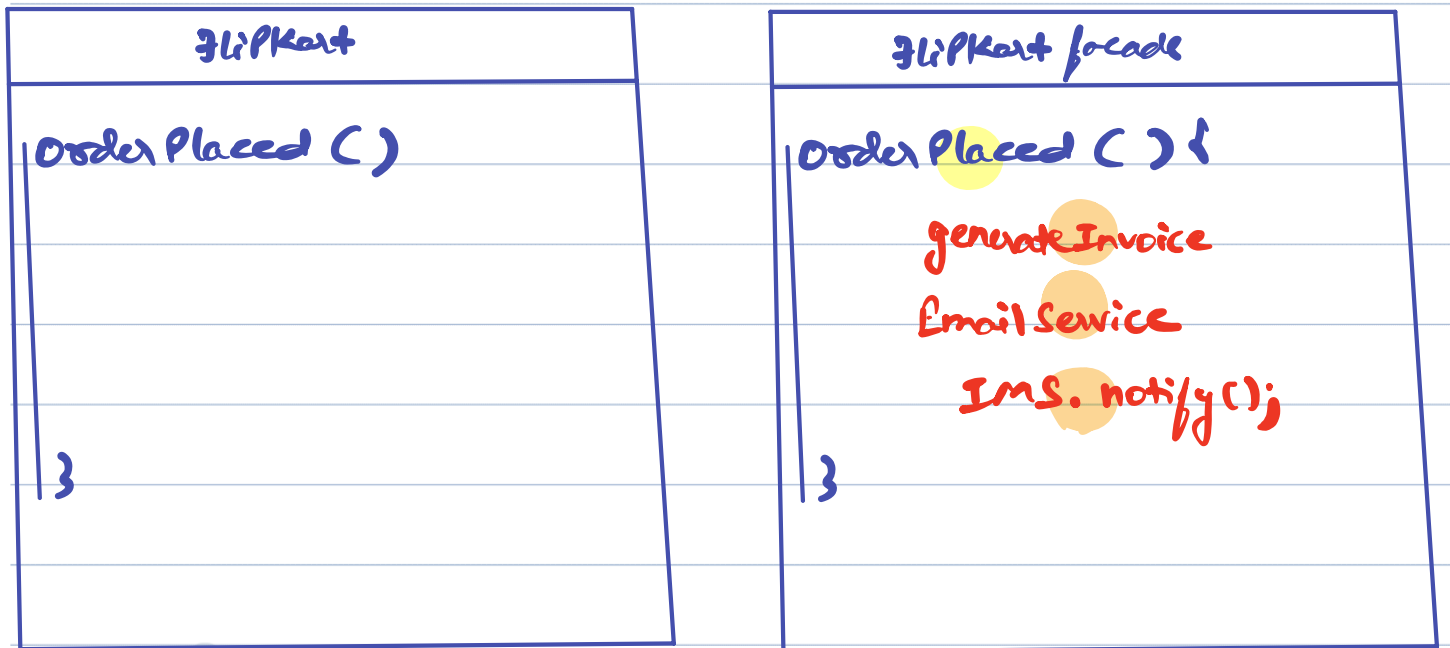
Abstract



AlgoPrep



Observer design Pattern

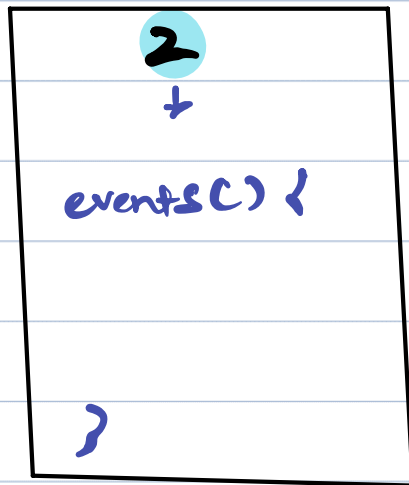


↳ when event happens and we might need to trigger multiple services but the services can be different for different scenarios.

↳ you can't add/remove these actions everytime.

↓
Observer design Pattern solves this problem.

1:m mapping



↑
Publisher
{selects events}

A

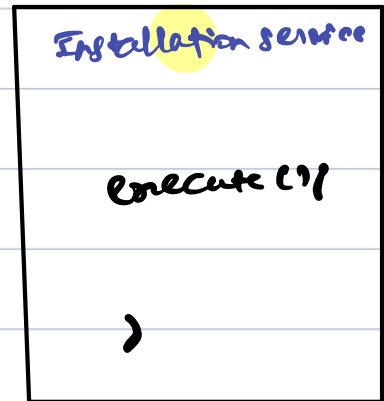
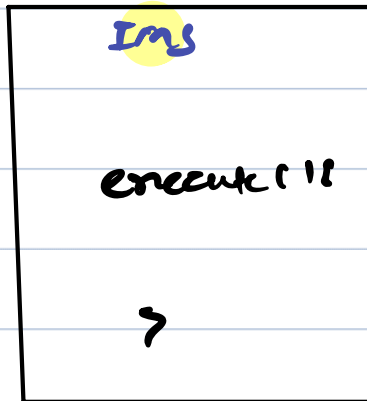
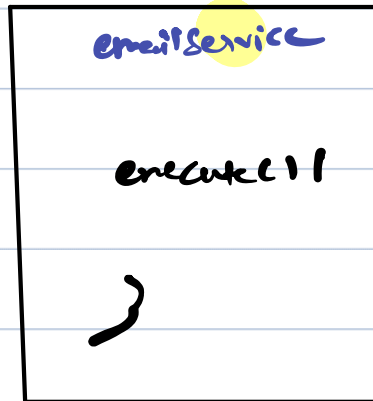
B

C

D

⇒ Subscribers
{services used
by publishers}

<<orderPlacedSubscriber>>





```
flipkart orderPlaced  
→ List < orderPlacedSubscribers > ops  
  
registerPlacedOrderSubscriber (orderPlacedSubscribers obj) {  
    ops.add(obj);  
}  
  
registerPlacedOrderSubscriber (orderPlacedSubscribers obj) {  
    =  
}
```

```
orderPlaced() {  
    for (OrderPlacedSubscribers temp: ops) {  
        temp.execute();  
    }  
}
```

→ At runtime, if you want to have dynamic things selected. you are looking for observer design pattern.