Adapter and Facade Design Pattern

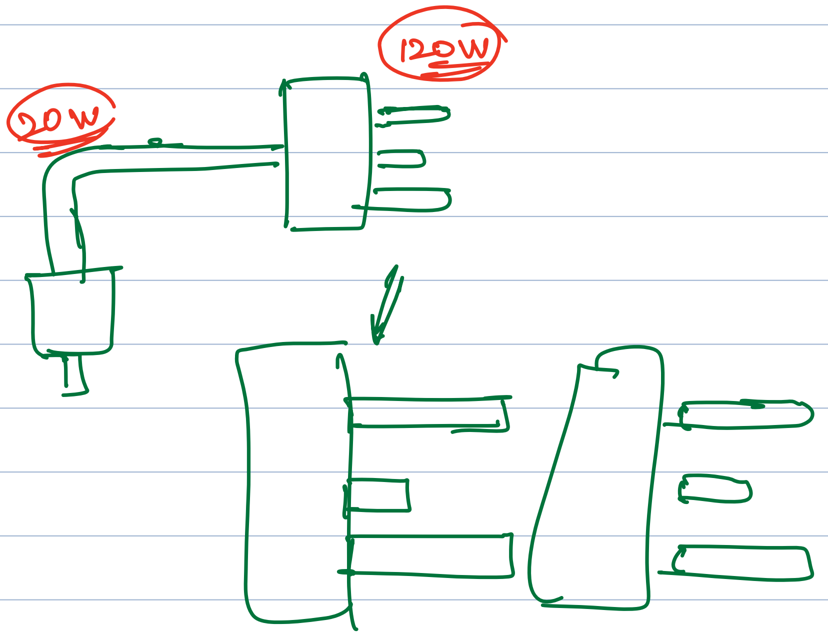
Agenda

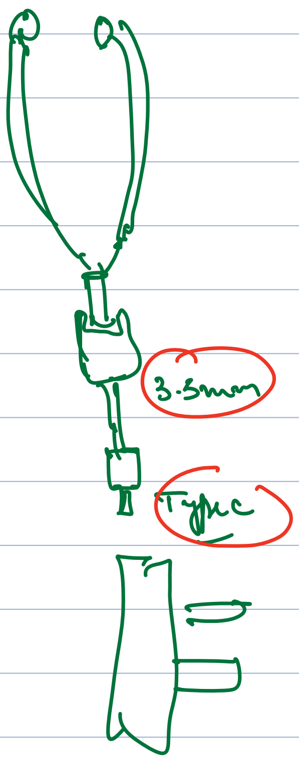
1. Intro to Structural Design Patterns
2. Adapter Design Pattern
3. Facade Design Pattern

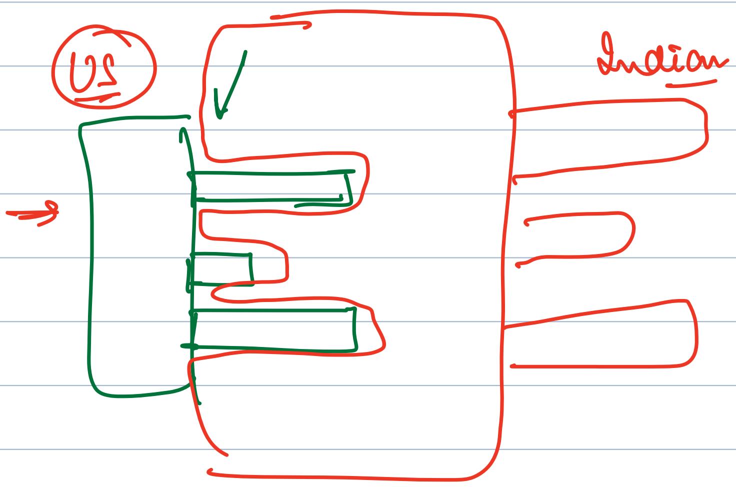
Structural Design Patterns

how to structure your codebase.
→ what classes to be there
→ what attr in every class
→ how should one class talk to another class

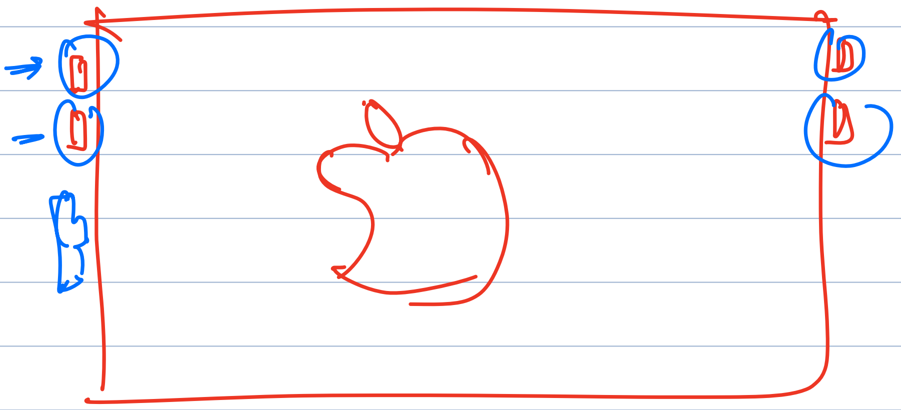
I Adapter Design Pattern







Adapter :→ ① intermediary layer.
② transform one form to another.

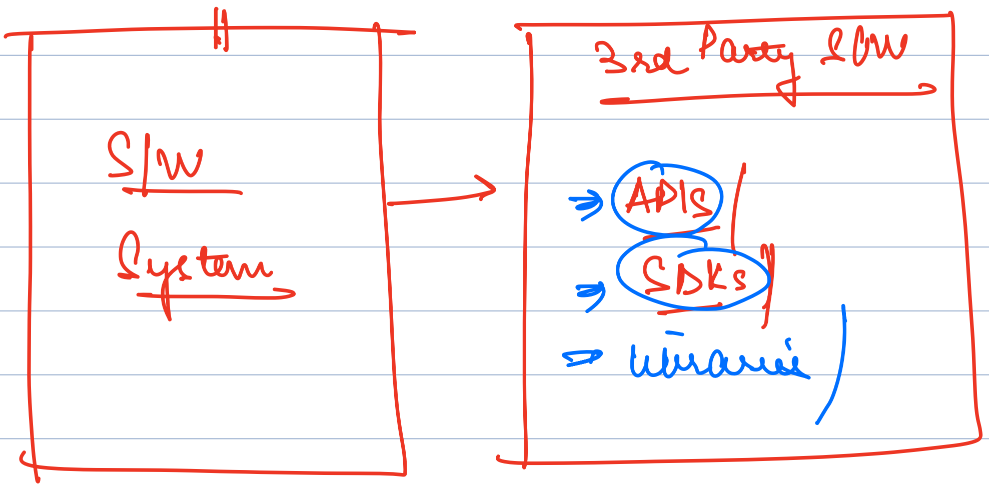


→ If apple was supporting multiple types of ports, their engineers would have to write code to send / receive data from each of the type. / understand diff types of data formats.

⇒ Currently, apple only supports one type of port (USBC) so their eng would have to write code only to support USBC data format.

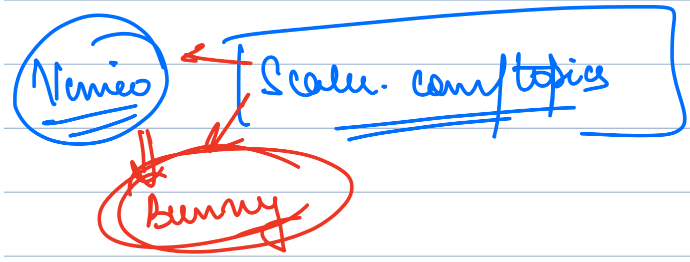
⇒ If anyone else wants to connect any other kind of device, adapter has to convert data from that format to type C format.

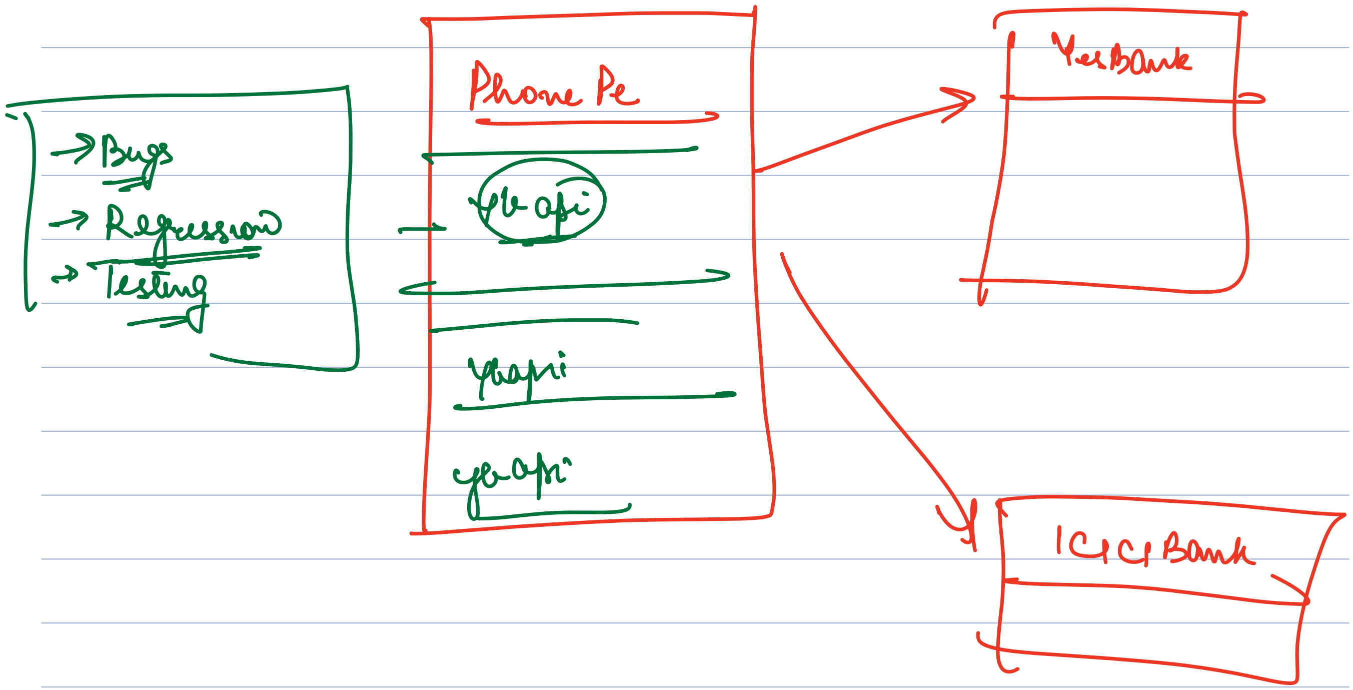
Adapter Design Pattern



Problem Statements

① You might want to change the 3rd party provider in future.





```
Phone Pe {
        Yes Bank  yb;


        Phone Pe ( Yes Bank  yb)   =
            this . yb = yb







}
```

Solⁿ ⇒ Adapter Design Pattern

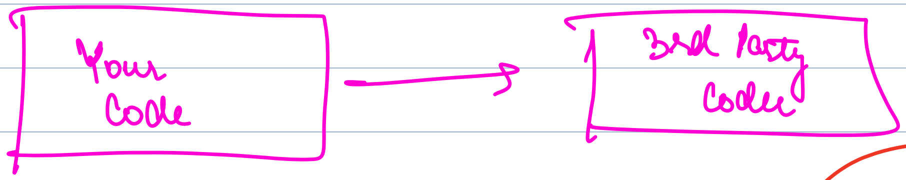
→ ensures that our codebase remains easily maintainable when its dependent on 3rd party APIs

① Should our codebase directly talk to 3rd Party Libraries ✗ No

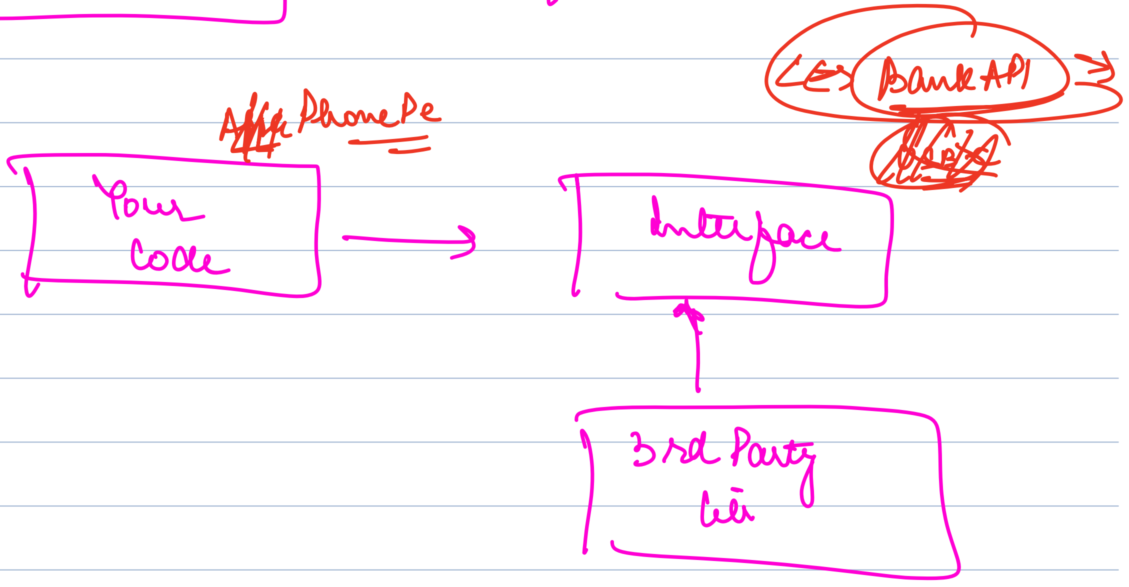
→ it will lead to tight coupling

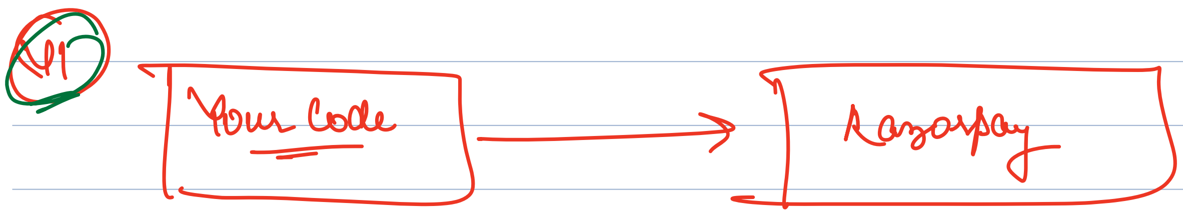
⇒ we should talk via an interface.

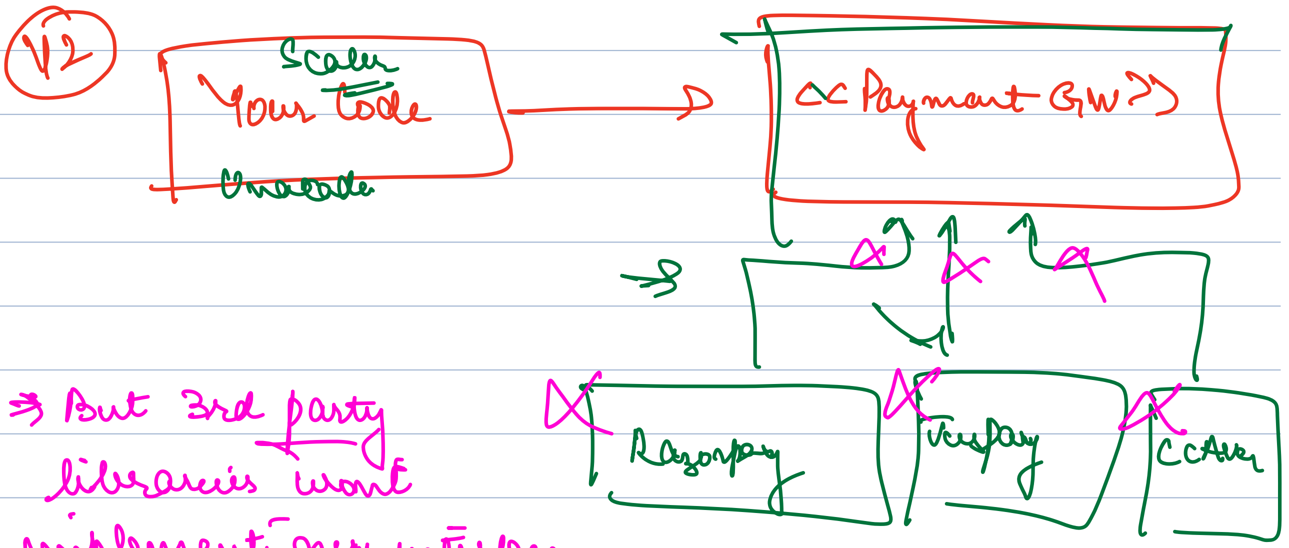
V1



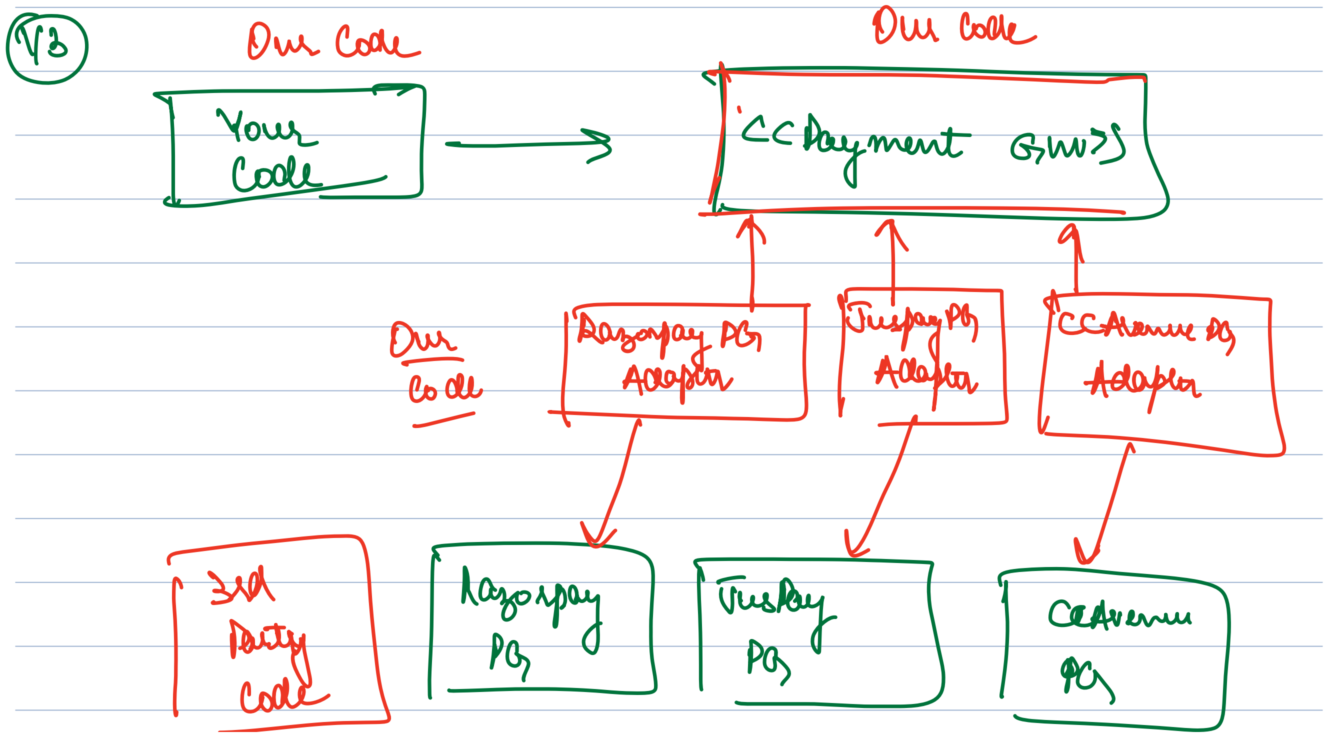
V2







⇒ But 3rd party libraries won't implement my interface



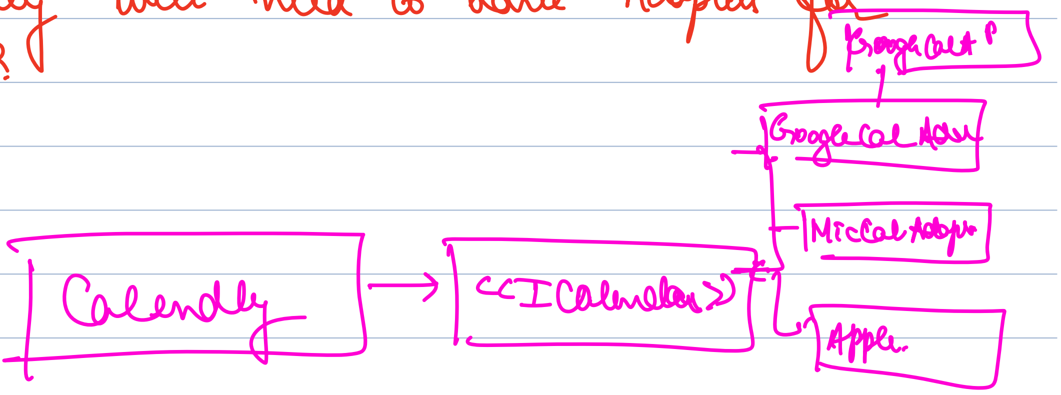
1. find out your req
2. for each req put method in interface
3. Figure out how to implement that via 3rd party

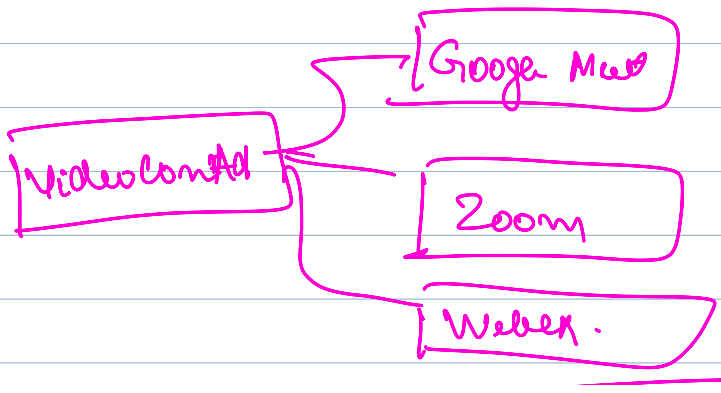
When to use Adapter ⇒ whenever talking to 3rd party API

Assignment

Calendly ⇒ Calendar Scheduling Platform

⇒ Calendly will need to have Adapter for what?





```
PaymentGWFactory {

        public static PaymentGW getPGForType
                              ( PGType type ) )
              if ( type == Razorpay )

    }
}
```