Bismillahir Rahmanir Rahim

In the name of Allah, the most gracious the most merciful

Welcome to the '**ADDA**' on

Adapter Pattern

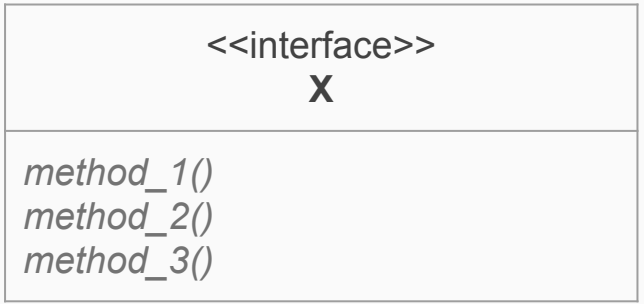
aka Wrapper Pattern

Adapters in real life

Adapter wraps something to represent it according to the client’s need.

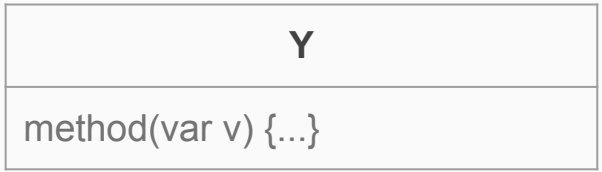
Adapters in software engineering

Client expects an implementation of ‘X’



We have something like that, but implementing ‘X’ doesn’t make enough sense.

Or, We may not have the source code of ‘Y’.

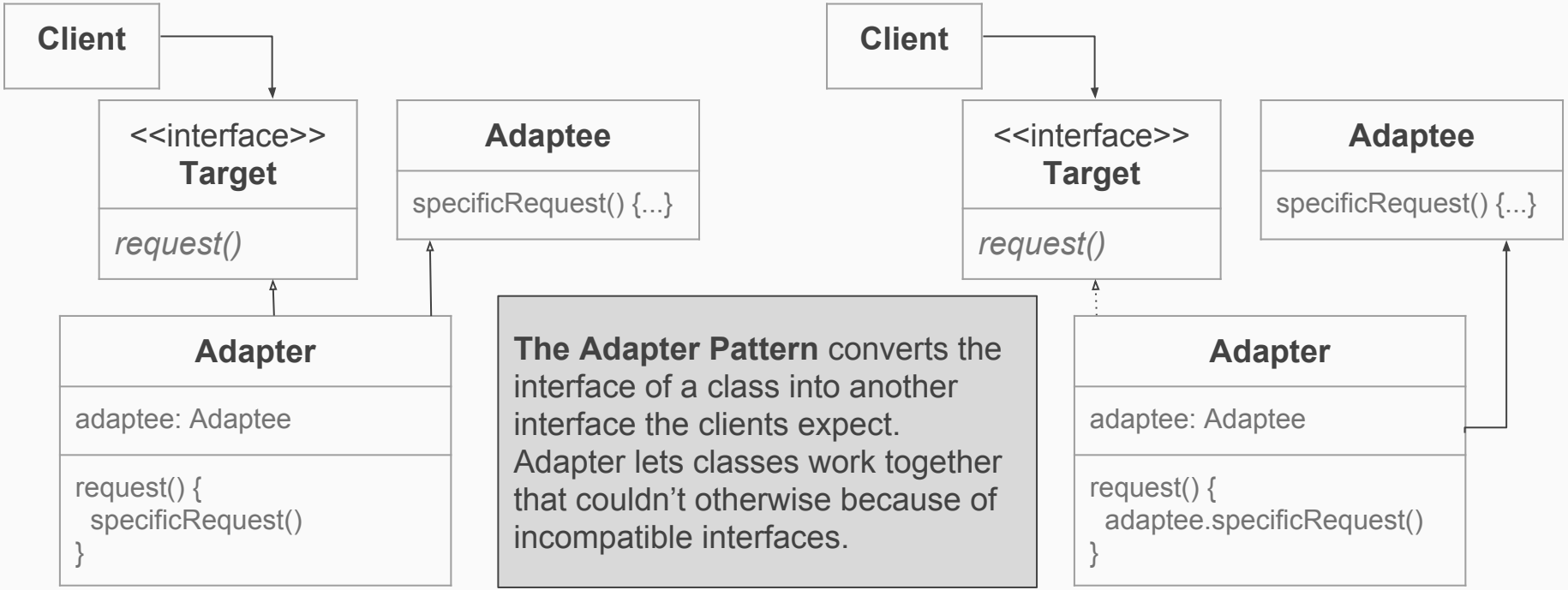


Then we should rather use an adapter.

Two options in hand

Class Adapter: Extends both ‘X’ and ‘Y’.

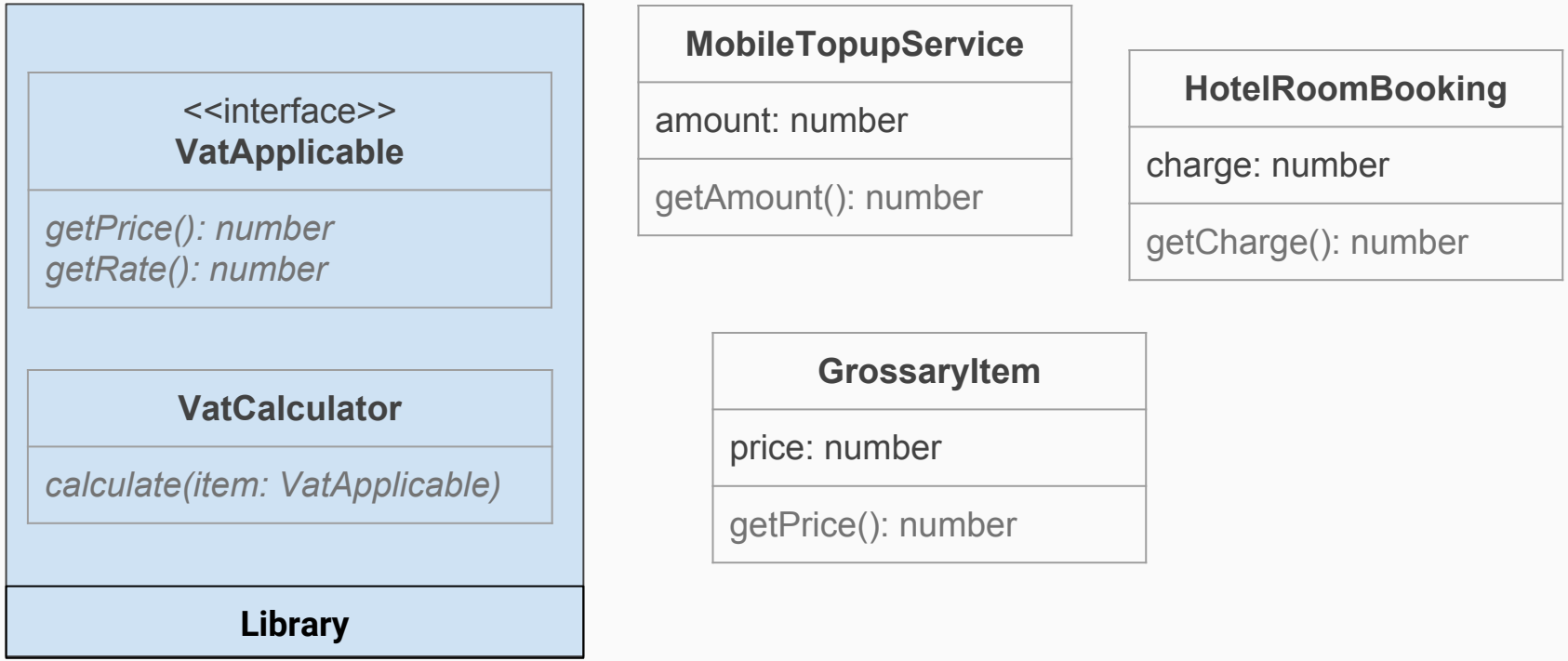
Object Adapter: Implement ‘X’ and compose ‘Y’.



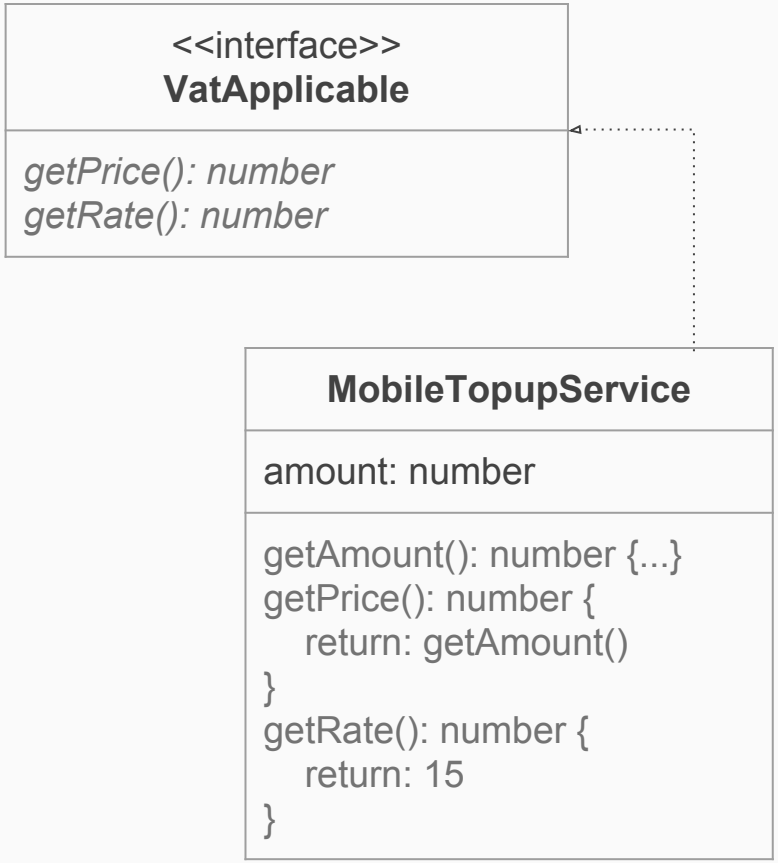
Use this pattern when -

- We want to use a class, but it doesn’t implement the interface we need.
- We want to create a reusable class which will cooperate with different unrelated classes.

Example


<<interface>>
VatApplicable
getPrice(): number
getRate(): number
VatCalculator
calculate(item: VatApplicable)
Library
MobileTopupService
amount: number
getAmount(): number
HotelRoomBooking
charge: number
getCharge(): number
GrossaryItem
price: number
getPrice(): number

Implement VatApplicable



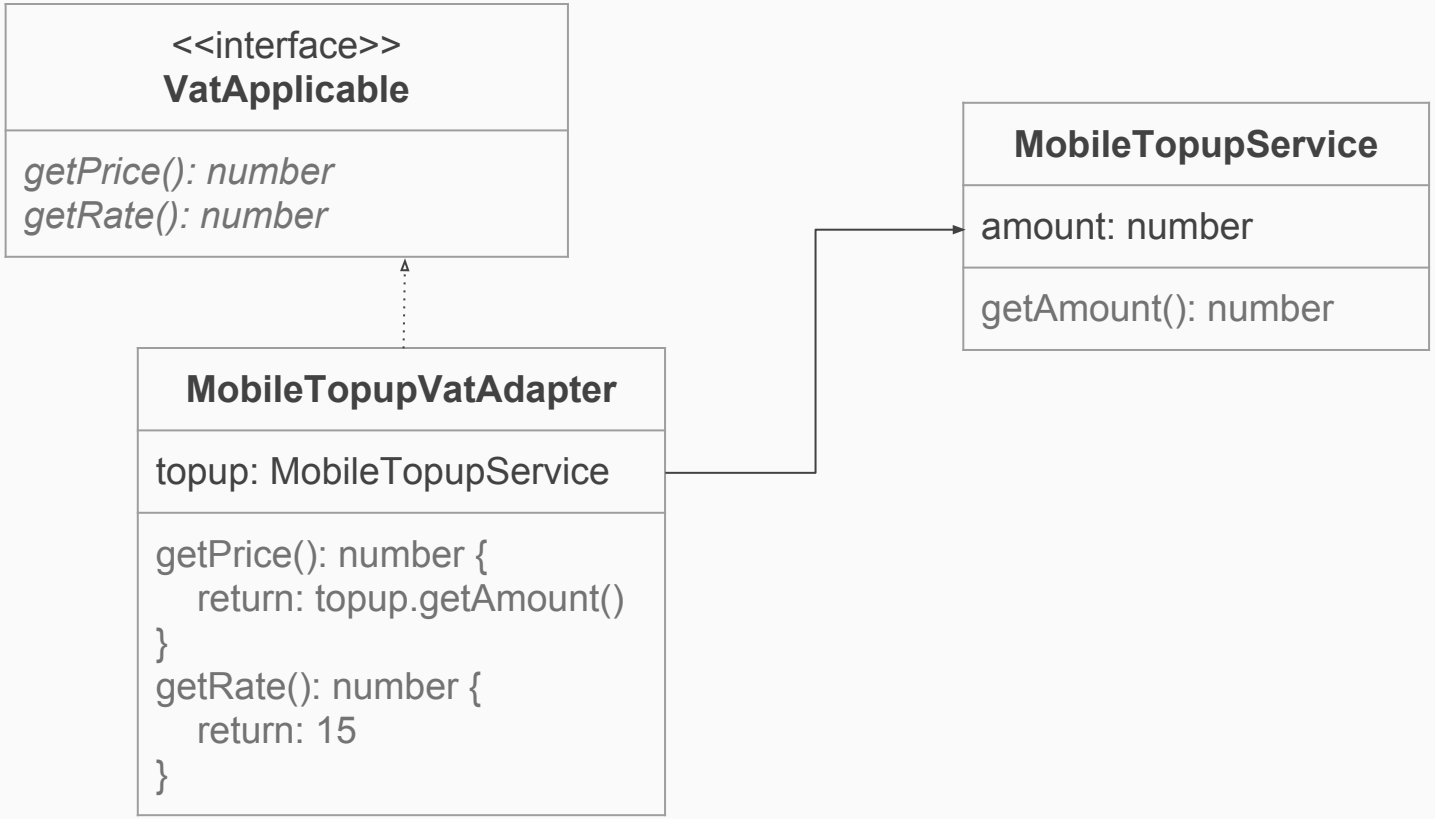
Problems

- Two methods doing same job.
- getRate method in MobileTopupService looks irrelevant.

So... Implement 'VatApplicable' is not a better solution.

We should rather use Adapter Pattern.

Use Adapter Pattern



Any other Real Life Use Cases?

Let’s share our ideas