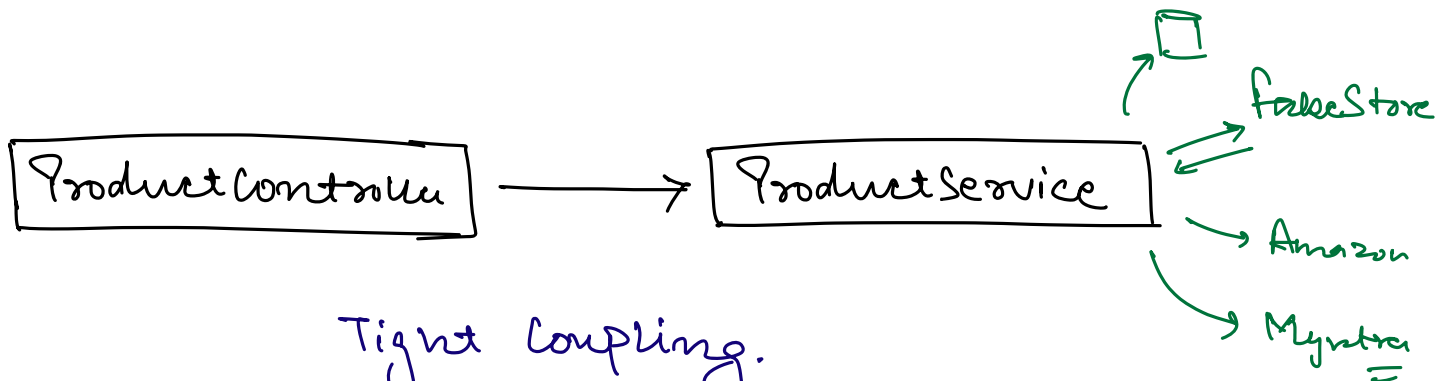


Agenda.

→ Implementing API's with fakeStore.

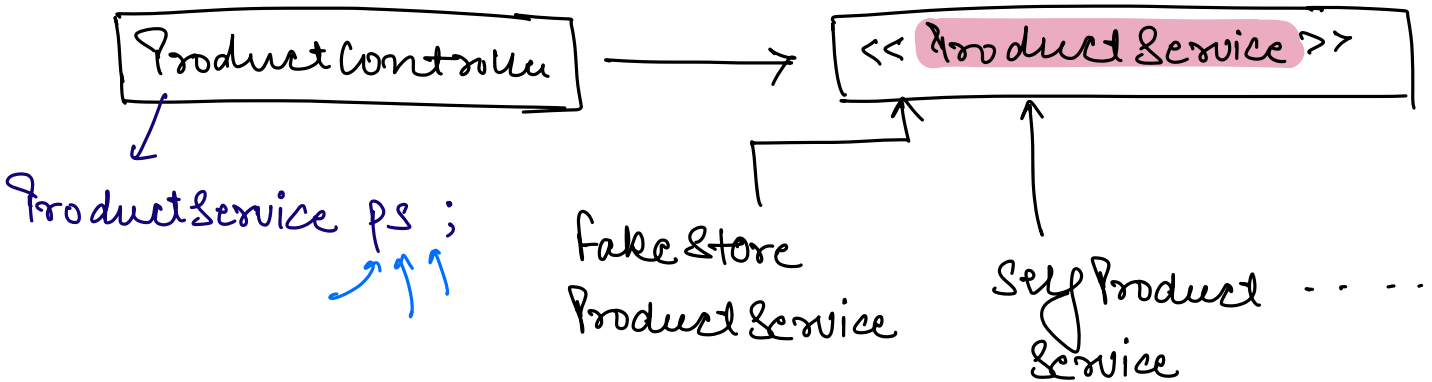
⇒



Tight Coupling.

⇒ There can be multiple ways of implementing ProductService class in future.

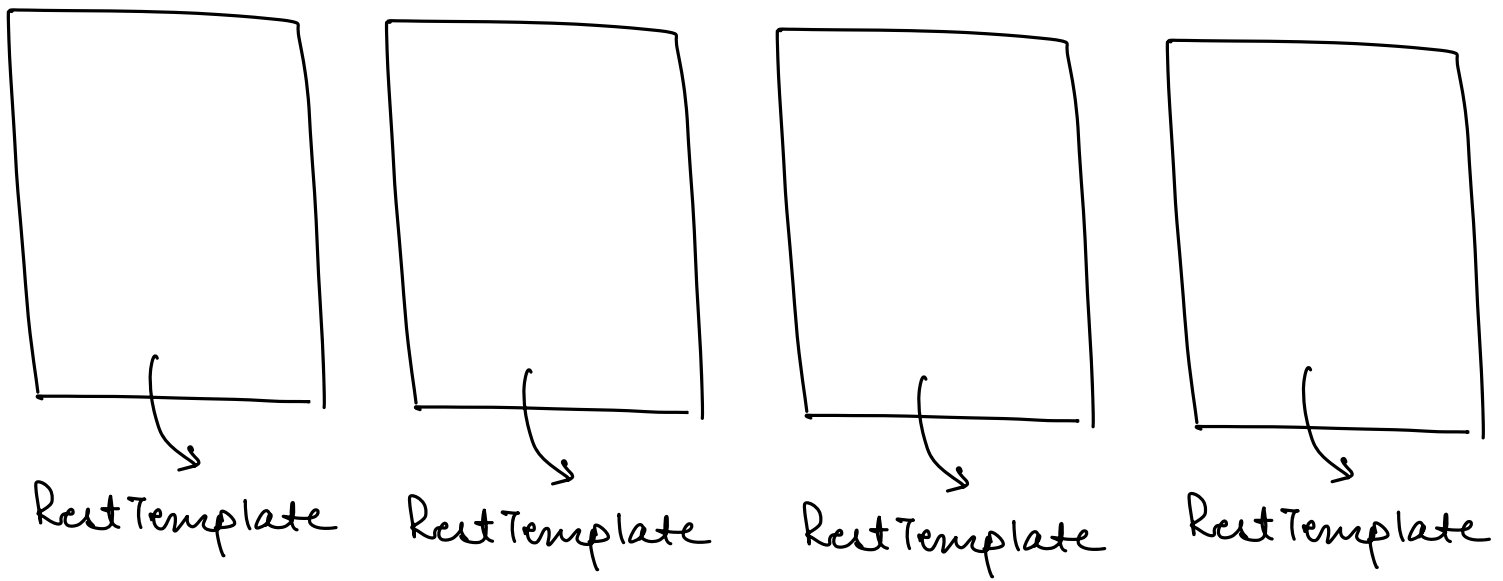
Interfaces.



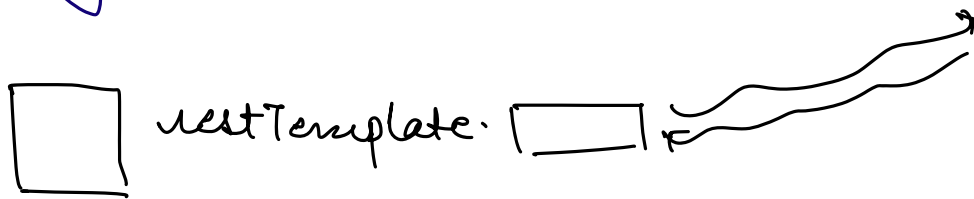
Ⓟ ← FakeStore ProductService

ps.findProduct(-)

⇒ Runtime Polymorphism.



⇒ Too many RestTemplate objects in the app.



⇒ Create a RestTemplate object at a common place and use it wherever required.

Spring Container |
Application Context.

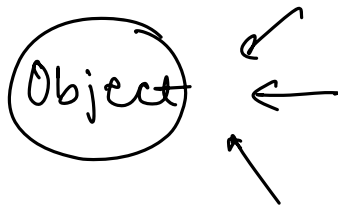
⇒ Bean

⇒ Create a **bean** of RestTemplate, store it inside the Spring Container and use it whenever required.

⇒ List<Animal>

List<Car>

List<User>



List<T>

Map<K, V>

↳ Generics

⇒ Generics were introduced in JAVA 5.

⇒ Before JAVA 5 :-

Raw Type.

List (list) = new ArrayList();

`list.add("Scaler")` ✓

`list.add(new Car())` ✓

`list.add(new Student())` ✓

⇒ No type safety.

`list < T >`
 ↑
 Placeholder
 ↗ `add(T)`
 → `T get(index)`

`list < Student > list = new ArrayList < > ();`

`list.add(new Student());`

`list.add(new Car())` ✗

⇒ Compile Time Error.

⇒ Type safety. ✓

⇒ JAVA is a backward compatible language.

⇒ After JAVA. 5

list list = _____
list<Animal> list2 = _____
list<Car> list2 = _____
↓
Compile ✓
↓
Run

⇒ Type Erasure

⇒ At runtime, every datatype gets erased and gets converted into Object