2_springCore-basic

https://www.notion.so/miraebyraejin/2-_-59ddc19355d646829c3d3fa9137d98fe

Lecture by Kim Younghan (inflearn.com/course/스프링-핵심-원리-기본편)

Personal documents

```
|Documents
|--classes.xlsx
|--diagram.drawio
|--readme.md
```

Class list: with fields and methods

Diagram: contains Domain relation, Class diagram, Object diagram

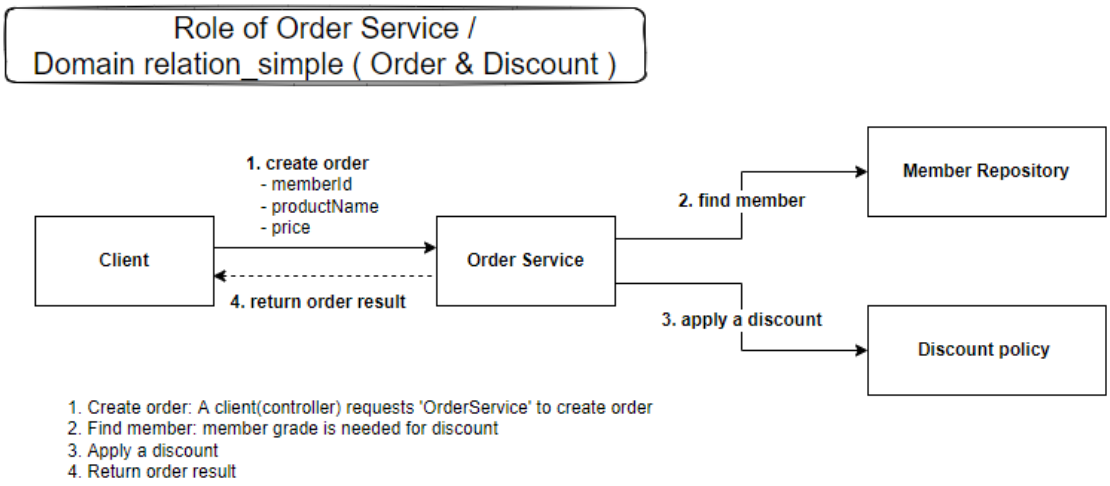
readme: this

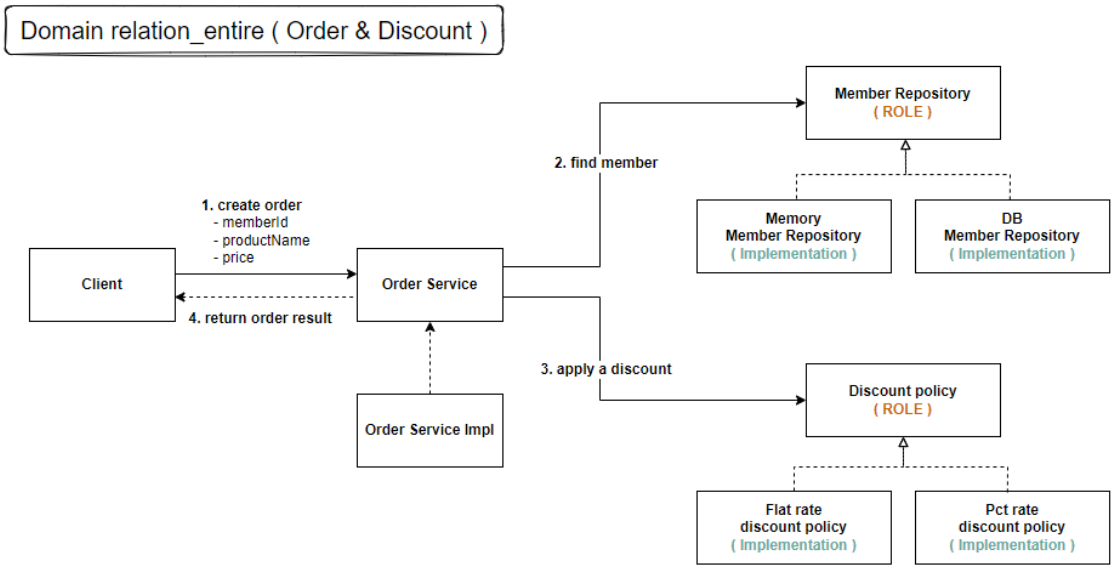
Summary of Main Points

▼ Distinguish role from implementation

Interface must focus only on its role(execution)

→ Role and Implementation can be separated by DI/IoC



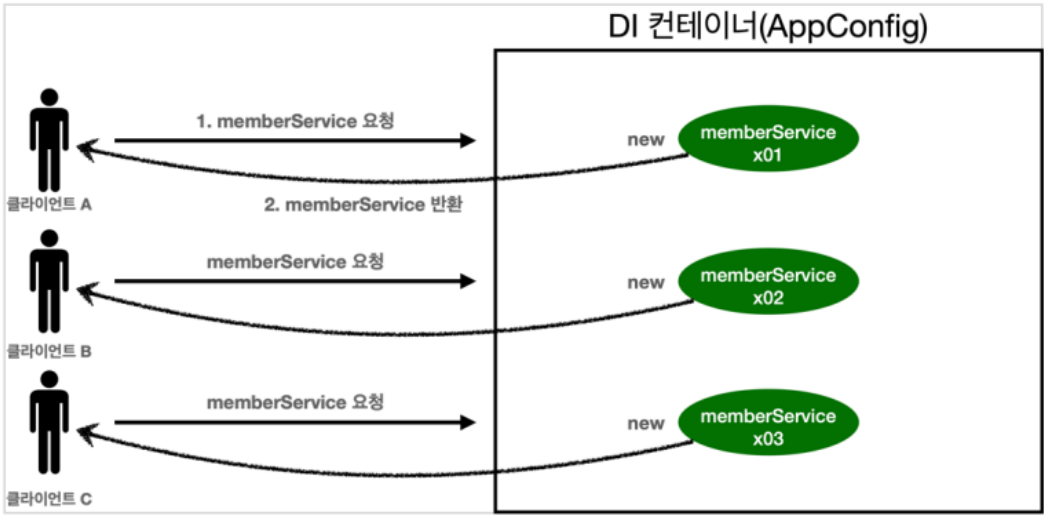


▼ DI Container / IoC Container

- XxxConfig.class

Here, create implementation objects and inject(link) them

However, it cause huge memory wasting because each objects are created every single call



→ Singleton pattern resolve this problem by creating and using only one object

▼ Singleton Pattern

It is regarded as Anti pattern because

Singleton pattern with plain java violates DIP, OCP and contains many other problems.

→ Singleton(Spring) Container can resolve this violation.

(Details: PDF - page 71 - 싱글톤 패턴 문제점, google)

- DIP violation

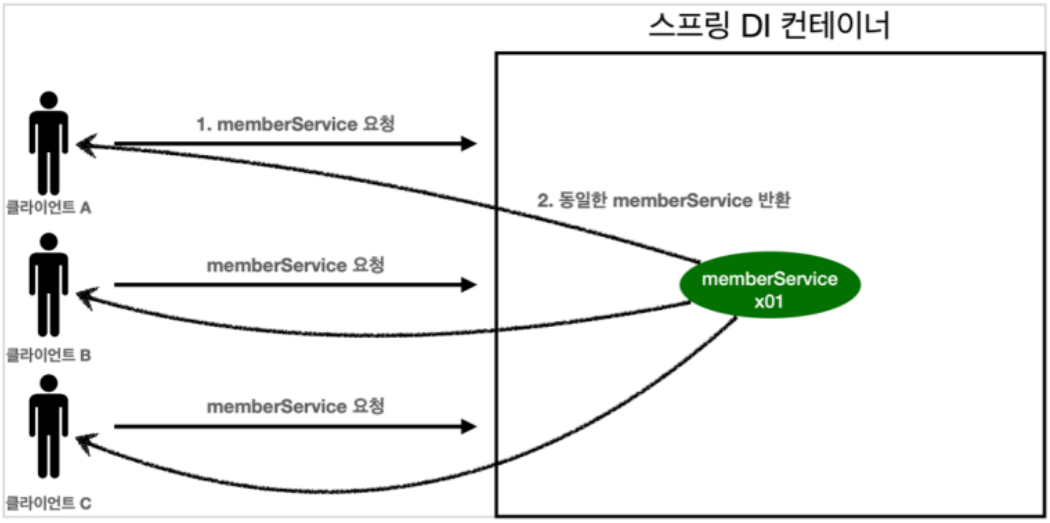
  Need to get instance from implementation class

  → depend on implementation not interface → It leads to OCP violation

```
public class AppConfig {
    return new MemberServiceImpl.getInstance();
}
```

▼ Spring Container / Singleton Container

→ ApplicationContext



- XxxConfig.class

  @Configuration

  Apply this annotation to XxxConfig.class to create as a spring container

  @Bean

  If exists, return the bean. Or else, enroll one.