Ministry of SMEs and Startups × KOSME Korea SMEs and Startups Agency × LIKELION

Dependency Injection

Nguyễn Anh Tuấn

K TECH COLLEGE

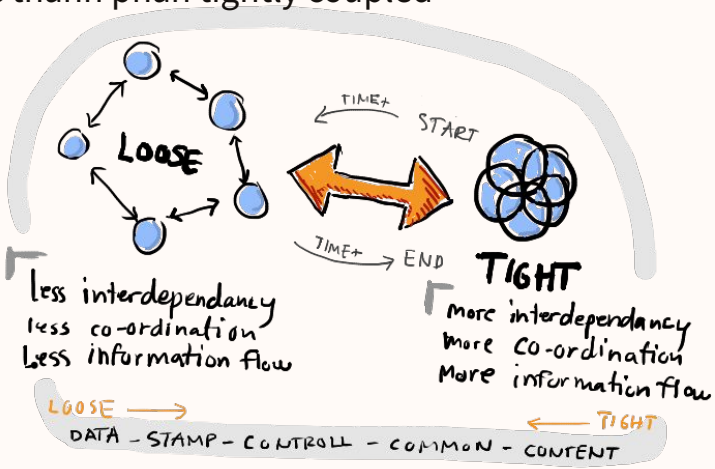
Nội dung bài giảng

1. Tight coupling và Loose coupling
2. Dependency injection (DI) là gì?
3. Các cách thực hiện DI

Tight coupling và Loose coupling

Tight coupling và Loose coupling

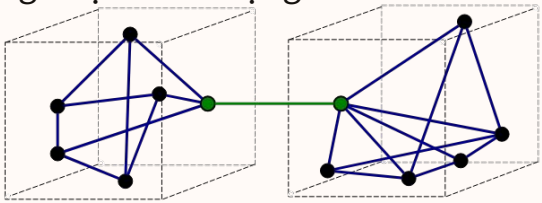
- **Tight coupling (liên kết ràng buộc)** xảy ra khi các lớp trong hệ thống có mối quan hệ phụ thuộc mạnh mẽ vào nhau, sự thay đổi trong 1 lớp có thể ảnh hưởng đến toàn bộ hệ thống hoặc các lớp khác.
- Nhược điểm của **Tight coupling**:
  - **Khó bảo trì**: Cần phải điều chỉnh nhiều lớp khác của hệ thống khi có 1 lớp thay đổi.
  - **Khó mở rộng**: Thêm hoặc thay đổi chức năng có thể gây ra nhiều vấn đề phụ thuộc.
  - **Khả năng tái sử dụng thấp**: Trong nhiều ngữ cảnh, các thành phần tightly coupled thường không thể được tái sử dụng.



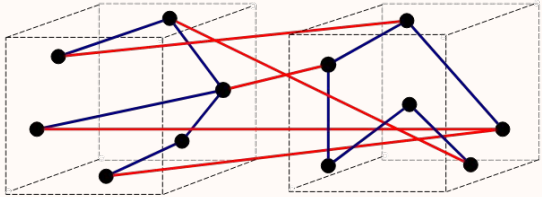
Tight coupling và Loose coupling

- **Loose coupling (liên kết lỏng)** giảm bớt sự phụ thuộc giữa các lớp với nhau. Trong **loose coupling**, các lớp hoạt động độc lập, không biết gì về cấu trúc của lớp khác.

- Ưu điểm của **Loose coupling**:
  - **Dễ bảo trì:** Thay đổi trong 1 lớp ít ảnh hưởng đến các lớp khác.
  - **Dễ mở rộng:** Thêm chức năng mới mà không ảnh hưởng đến hệ thống hiện tại.
  - **Khả năng tái sử dụng cao:** Các thành phần loosely coupled dễ dàng được tái sử dụng trong các ngữ cảnh khác nhau.



a) Good (loose coupling, high cohesion)

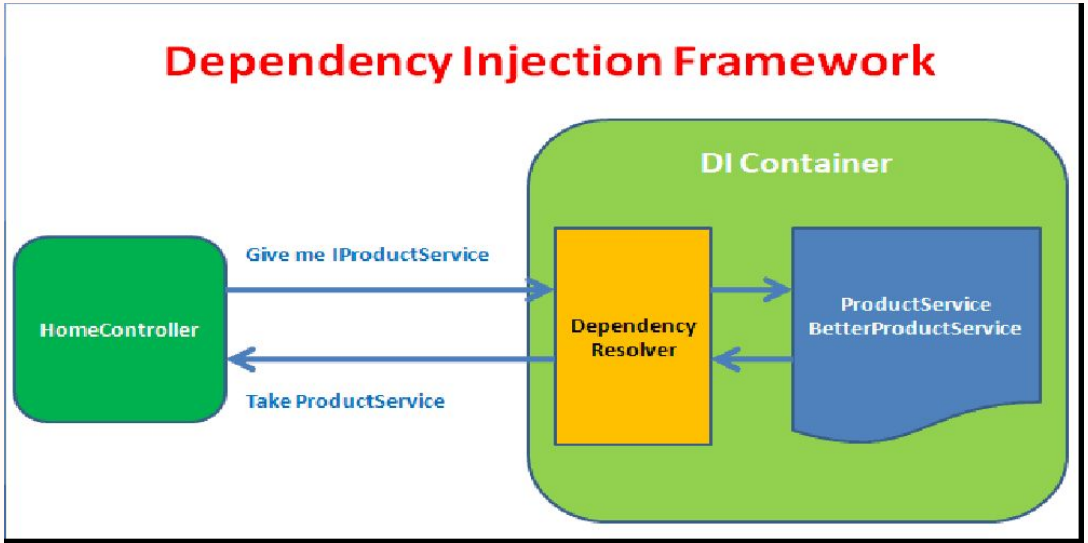


b) Bad (high coupling, low cohesion)

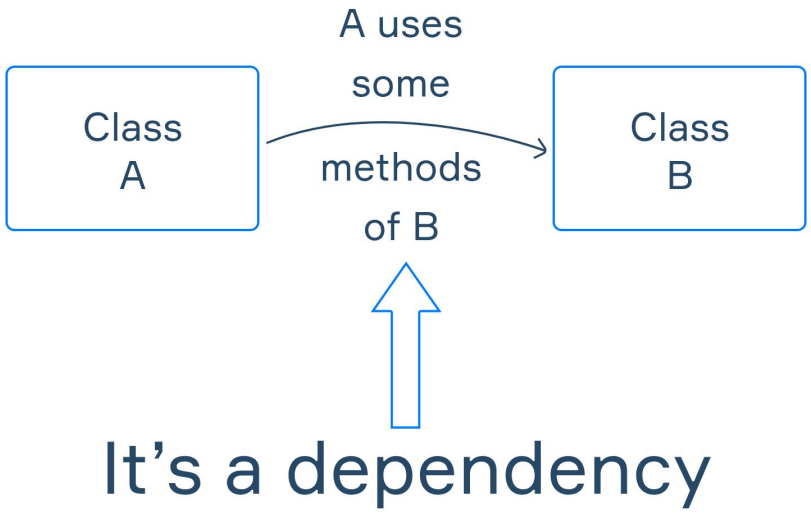
Dependency Injection

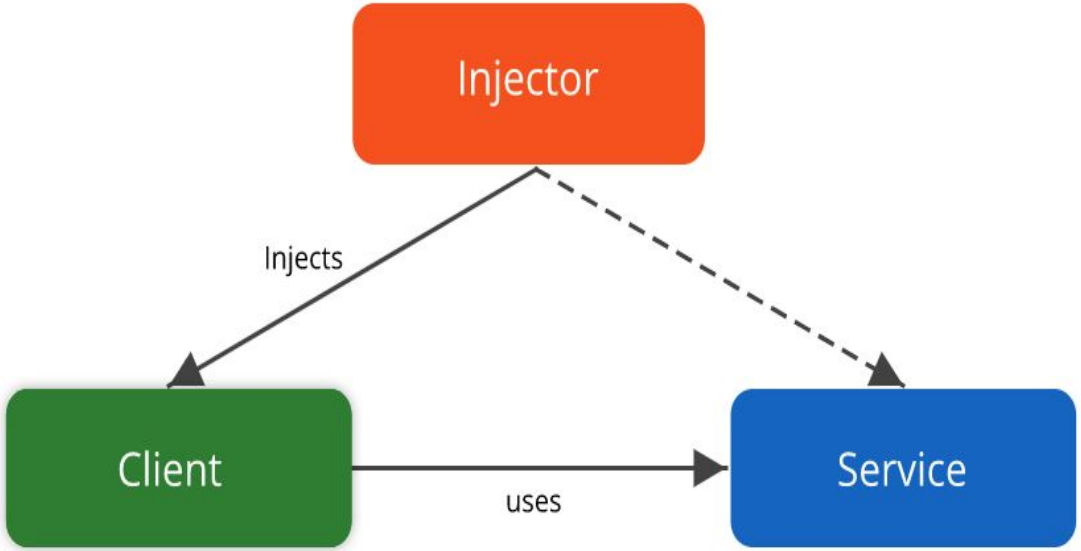
Dependency Injection

- **Dependency Injection** là 1 kỹ thuật, 1 mô hình lập trình (design pattern) cho phép xóa bỏ sự phụ thuộc hard-code và làm cho ứng dụng dễ mở rộng và maintain hơn.
- **Dependency Injection** không chỉ áp dụng cho Java mà còn cho nhiều ngôn ngữ lập trình khác.
- Trong **Dependency Injection**, các phụ thuộc của 1 lớp không được tạo bên trong lớp đó, mà nó được truyền vào từ bên ngoài.



Dependency Injection





Dependency Injection

- **Ưu điểm**
  - Giảm sự kết dính giữa các module
  - Code dễ bảo trì, dễ thay thế module
  - Rất dễ test và viết Unit Test
  - Dễ dàng thấy quan hệ giữa các module (Vì các dependency đều được inject vào constructor)

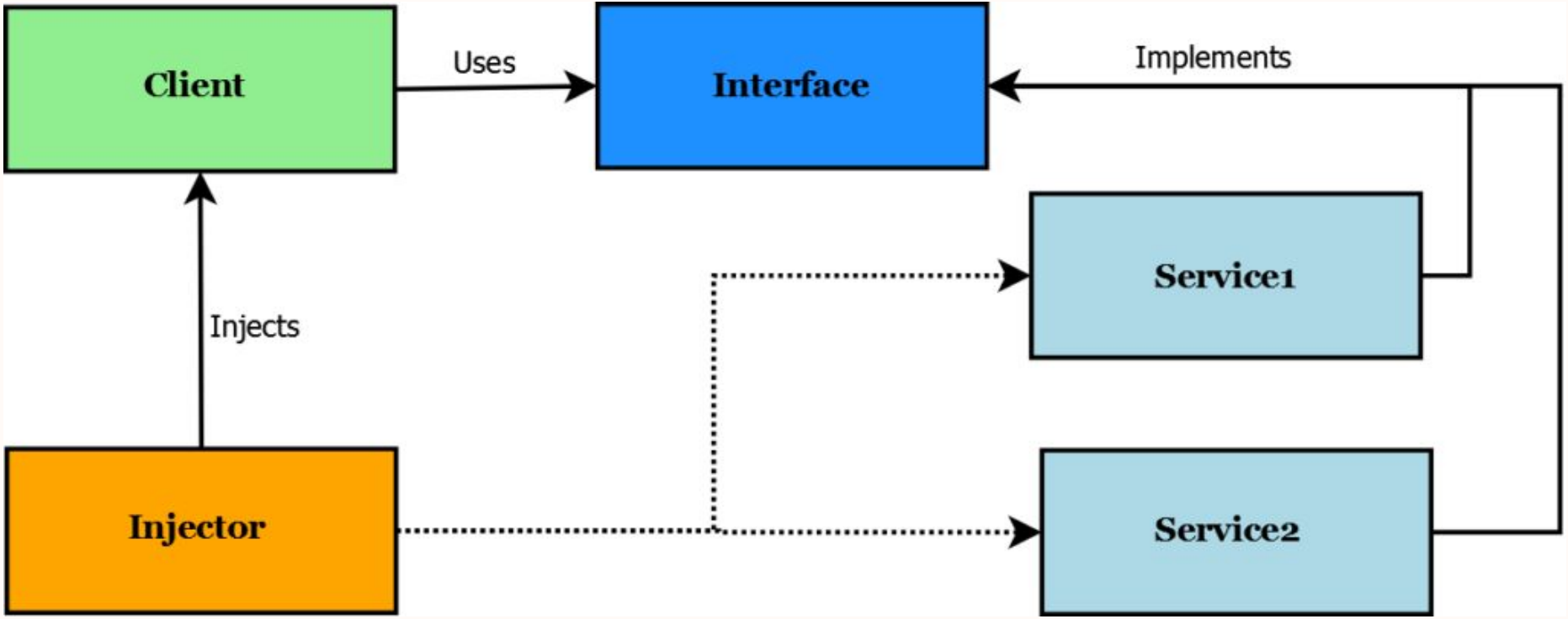
- **Nhược điểm**
  - Khái niệm DI hơi khó hiểu với người mới
  - Khó debug vì không biết implements nào của interface được gọi đến
  - Các object được khởi tạo từ đầu làm giảm performance
  - Làm tăng độ phức tạp của code

Các cách thực hiện DI

Các cách thực hiện Dependency Injection

- Các cách thực hiện **Dependency Injection**:
  - **Constructor Injection:** Các dependency sẽ được truyền vào (inject vào) 1 class thông qua constructor của class đó.
  - **Setter Injection:** Các dependency sẽ được truyền vào 1 class thông qua các hàm Setter/Getter.
  - **Interface Injection:** Các dependency sẽ được truyền vào 1 class thông qua việc ghi đè lại phương thức của interface.
  - **Public Field Injection:** Các dependency sẽ được truyền vào 1 class một cách trực tiếp vào các public field.

Dependency Injection



ROAD TO KOREA

Nếu có bất kỳ thắc mắc nào, hãy đặt câu hỏi qua

mail@mail.com hoặc Zalo 0xxx xxx xxx