Bài tập bắt buộc của Tuần XX GV sẽ công bố cuối giờ thực hành. Bài tập bắt buộc sẽ gửi với Subject và tập tin nén kèm theo:
LTWWW_JAVA_{NGAYTHANGNAM}_TUAN{XX}_HOTENSINHVIEN

(trong đó XX sẽ là 01, 02 ... 10).

VD: LTWWW_JAVA_20082024_TUAN01_HOTENSINHVIEN

Sinh viên nộp sai yêu cầu bài tập Tuần nào xem như không nộp bài tập Tuần đó.

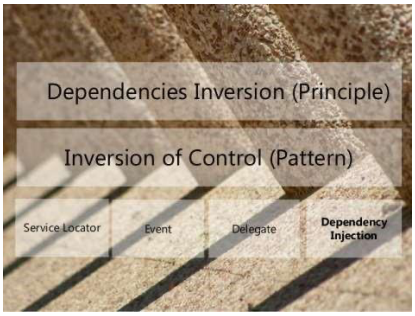
BÀI TẬP TUẦN 5 MÔN LẬP TRÌNH WEB NÂNG CAO (JAVA)

1. ***Chương 4: SpringEcoSystem***

   Mục tiêu:

   - *Hiểu và hiện thực được Spring IoC và DI*
   - *Hiểu và hiện thực được Spring Bean*

SOLID Priciples



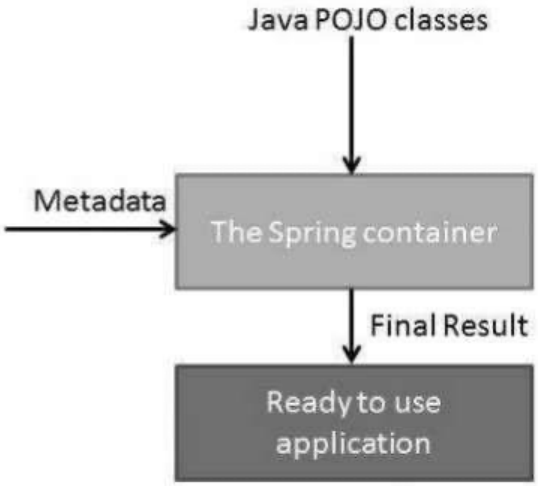
IoC Container trong Spring là core của Spring Framework. IoC Container, IoC tạo ra các đối tượng, nối các đối tượng lại với nhau, cấu hình, và quản lý life cycle (từ khi tạo ra đến khi bị hủy). IoC Container sử dụng DI để quản lý các thành phần tạo nên một ứng dụng. Các đối tượng này được gọi là Spring Bean.

IoC Container được cung cấp thông tin từ các tập tin XML.

Hai loại IoC containers:

- BeanFactory
- ApplicationContext

- ***ApplicationContext*** (***org.springframework.context.ApplicationContext*** interface) cũng tương tự như ***BeanFactory*** (***org.springframework.beans.factory.BeanFactory***), sử dụng để mô tả Spring Container. ApplicationContext được xây dựng trên interface BeanFactory.
- BeanFactory cung cấp các chức năng cơ bản trong khi ApplicationContext cung cấp các tính năng mở rộng hơn cho các ứng dụng Spring nên ApplicationContext phù hợp với các ứng dụng J2EE hơn. Lưu ý: ApplicationContext bao gồm tất cả các chức năng của BeanFactory.
- Đối tượng của interface ApplicationContext như là một khung chứa của Spring để gọi đến đối tượng cần lấy bằng cách sử dụng phương thức getBean().



Dependency Injection:

- Các module không giao tiếp trực tiếp với nhau, mà thông qua interface. Module cấp thấp sẽ implement interface, module cấp cao sẽ gọi module cấp thấp.

  Ví dụ: Để giao tiếp với database, sử dụng interface IDatabase, các module cấp thấp là MySQLDatabase, SQLDatabase, MariaDBDatabase, ... Module cấp cao là CustomerBusiness sẽ sử dụng interface IDatabase.
- Việc khởi tạo các module cấp thấp sẽ do DI Container thực hiện.

  Trong module CustomerBusiness, không khởi tạo IDatabase db = new XMLDatabase(), việc này sẽ do DI Container thực hiện. Module CustomerBusiness sẽ không biết gì về module XMLDatabase hay SQLDatabase.
- Các Module kết hợp với các interface cụ thể sẽ được config trong code hoặc trong file XML.

- DI được dùng để làm giảm sự phụ thuộc giữa các module, dễ dàng trong việc thay đổi module, bảo trì code và testing.

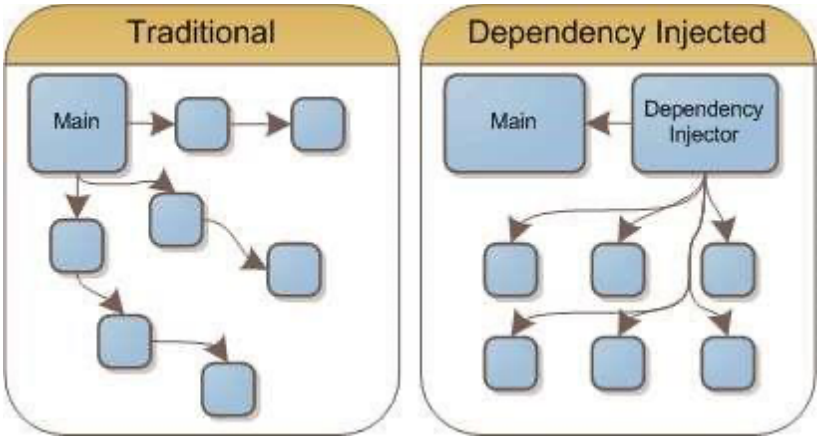
Thực hiện Dependency Injection theo 3 cách:

1. ***Constructor Injection***: Các dependency sẽ được container truyền/inject vào 1 class thông qua constructor của class.
2. ***Setter Injection***: Các dependency sẽ được truyền vào 1 class thông qua các hàm setter.
3. ***Interface/Method Injection***: Class cần truyền/inject sẽ implement 1 interface. Interface này có chứa 1 hàm tên Inject. Container sẽ injection dependency vào 1 class thông qua việc gọi hàm Inject của interface.

Bài tập 1:

▪ Làm việc với Dependency Injection
▪ Thao tác Spring Dependency Injection

IoC trong Spring và phương pháp tiếp cận truyền thống.



Làm lại tất cả ví dụ trong Slide bài giảng phần: Configuring Beans in the Container

Bài tập 2:

Một Nhân viên sẽ có các thông tin như họ tên, địa chỉ .... Thiết kế đối tượng Employee và Address dùng Dependency Injection (DI) trong Spring để loại bỏ sự phụ thuộc giữa các mã chương trình.

Tham khảo hướng dẫn:

B1. Tạo Project Java
B2. Configure Project Convert to Maven Project
B3. Cấu hình các dependency của Spring trong pom.xml

Dependencies

```
<!-- https://mvnrepository.com/artifact/org.springframework/spring-core -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-core</artifactId>
    <version>6.1.12</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-context -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>6.1.12</version>
</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-beans -->
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-beans</artifactId>
    <version>6.1.12</version>
</dependency>
```

B4. Tạo 2 classes

Address.java

```
public class Address {

    private String city;

    private String state;
```

```
    private String country;

    // Constructor

    // getter + setter

}
```

Employee.java

```
public class Employee {

    private int id;
    private String name;
    private Address address;

   // Constructor

  // getter + setter

}
```

B5:

- Dùng XML-Based Configuration: ***ClassPathXmlApplicationContext*** (xml file)

  Link hướng dẫn tạo file beans.xml

  https://docs.spring.io/spring-framework/docs/4.2.x/spring-framework-reference/html/xsd-configuration.html#xsd-config-body-referencing

  + Tạo bean sử dụng setter

  + Tạo bean sử dụng constructor
- Dùng Annotation-Based Configuration: ***AnnotationConfigApplicationContext***
- Dùng Java-Based