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

Spring IoC - Inversion of Control

Nguyễn Anh Tuấn

KTECH COLLEGE

Nội dung bài giảng

1. Spring IoC - Inversion of Control
2. @Component, @Autowired
3. Spring Bean, Bean Scope
4. @Configuration, @Bean, @Scope

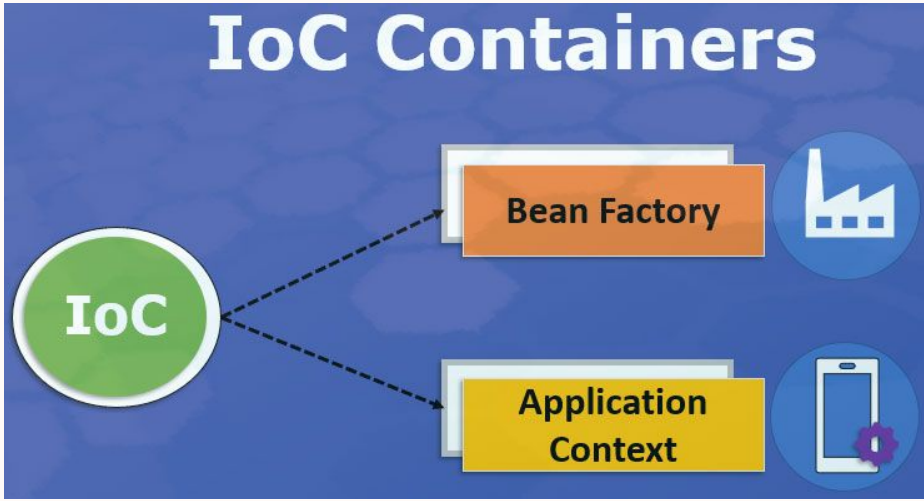
Inversion of Control

Spring IoC - Inversion of Control

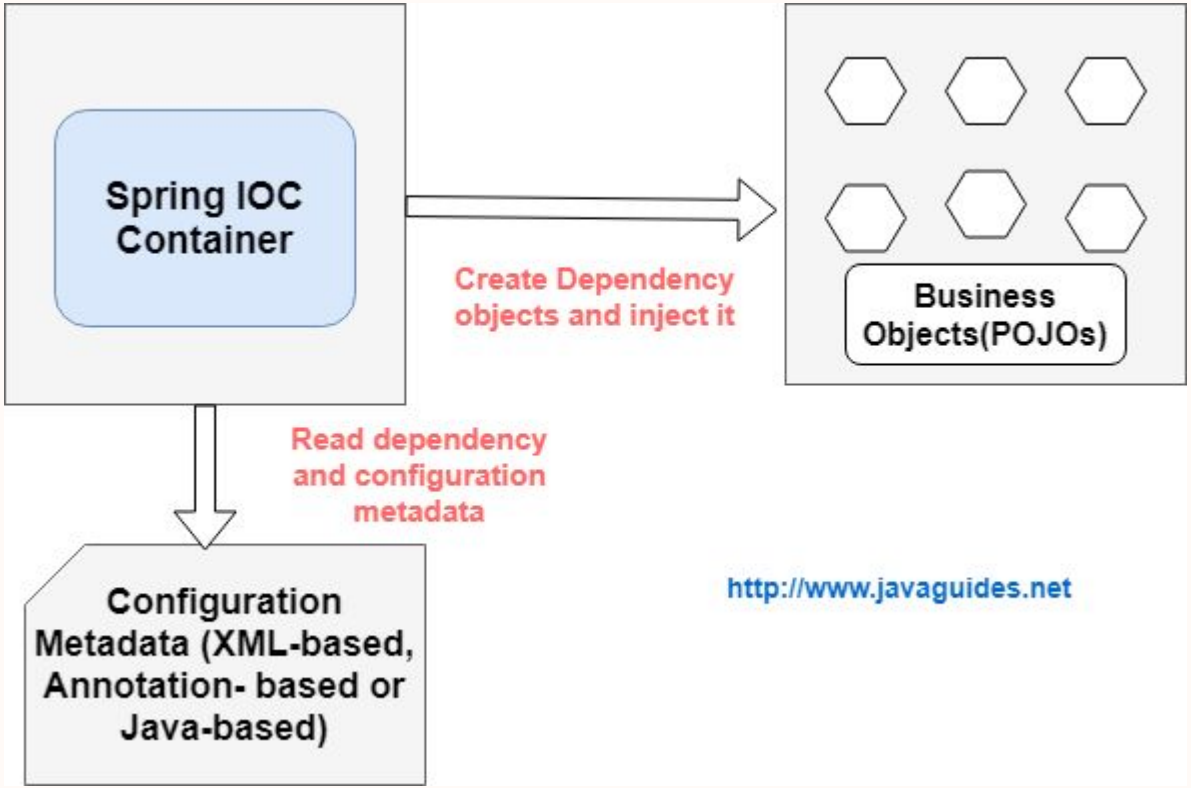
- **Inversion of Control (IoC)** là 1 nguyên tắc thiết kế, trong đó việc điều khiển các đối tượng hoặc các phần của chương trình được chuyển giao cho 1 container hoặc framework bên ngoài. Thay vì ứng dụng điều khiển luồng thực thi, container hoặc framework sẽ làm việc này.

- Spring Framework cung cấp 1 **IoC Container** để quản lý lifecycle của các đối tượng và phụ thuộc của chúng. Các container chính trong Spring là **BeanFactory** và **ApplicationContext**.

- **BeanFactory** là container cơ bản nhất cung cấp các chức năng cốt lõi.
  ApplicationContext là một extension của BeanFactory với các tính năng nâng cao hơn.



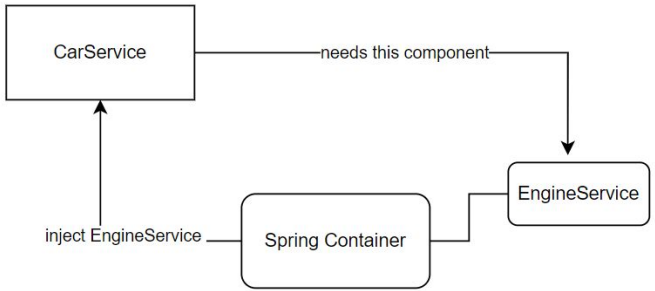
Spring IoC - Inversion of Control



@Component,
@Autowired

@Component, @Autowired

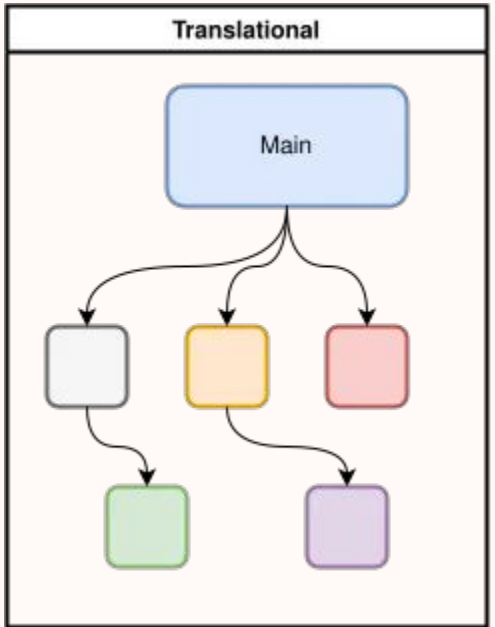
- **Annotation:** Là 1 tính năng quan trọng trong lập trình Java, cho phép bổ sung thông tin, đánh dấu và cung cấp metadata cho các lớp, phương thức, biến, package, giúp cho trình biên dịch hiểu, xử lý 1 cách thông minh, nhanh chóng.
- **@Component:** Là 1 annotation đánh dấu trên các class để cho biết class đó là 1 bean. Spring Boot sẽ tạo và quản lý các instance của class được đánh dấu là bean trong **IoC Container** để bất cứ khi nào cần sử dụng thì không cần khởi tạo lại instance mới.
- **@Autowired:** Là 1 annotation sử dụng để truyền (inject) các dependency vào các thành phần khác. Khi 1 thuộc tính được đánh dấu bằng **@Autowired**, Spring Boot sẽ tự động tiêm 1 instance của dependency tương ứng vào thuộc tính đó.

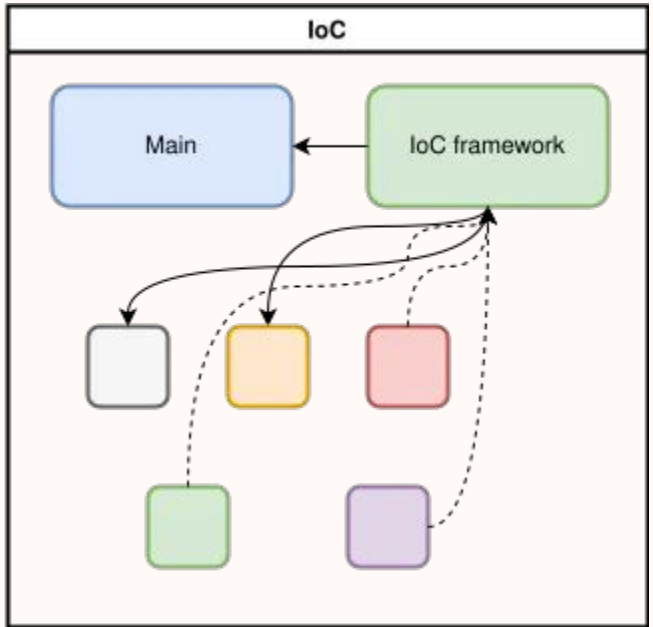


Spring Bean, Bean Scope

Spring Bean, Bean Scope

- **Spring Bean:** Những object được quản lý trong **Spring IoC Container** được gọi là **Bean**.
- **Bean** là những object được khởi tạo, lắp ráp (assembled) và quản lý bởi **Spring IoC Container.**





Spring Bean, Bean Scope

- **Cách tạo ra 1 Bean:**
  - **@Component:** Đánh dấu một lớp là một Spring bean. Các bean này sẽ được Spring container quản lý.
  - **@Controller:** Đánh dấu một lớp là một Spring MVC controller, chịu trách nhiệm xử lý các request HTTP.
    - Dùng trong các ứng dụng web để định nghĩa các endpoint xử lý request và trả về response.
  - **@Service:** Đánh dấu một lớp là một service, nơi chứa logic nghiệp vụ của ứng dụng.
    - Dùng để phân chia và tổ chức các tầng logic nghiệp vụ trong ứng dụng.
  - **@Repository:** Đánh dấu một lớp là một repository, chịu trách nhiệm tương tác với database.
    - Dùng để quản lý các thao tác CRUD và truy vấn dữ liệu từ database.

@Configuration, @Bean, @Scope

@Configuration, @Bean, @Scope

- **@Bean method:** Đánh dấu 1 phương thức trong lớp cấu hình (configuration class) là 1 bean factory method. Phương thức này sẽ trả về 1 đối tượng mà Spring IoC Container sẽ quản lý như một bean.
    - Dùng trong các lớp @Configuration để khai báo và cấu hình các bean.

- **@Configuration:** Là 1 annotation đánh dấu 1 class chứa các thông tin cấu hình cho ứng dụng.
- Spring Boot sẽ quét các class được đánh dấu là @Configuration để tạo và quản lý các phương thức được đánh dấu là Bean

Spring Bean, Bean Scope

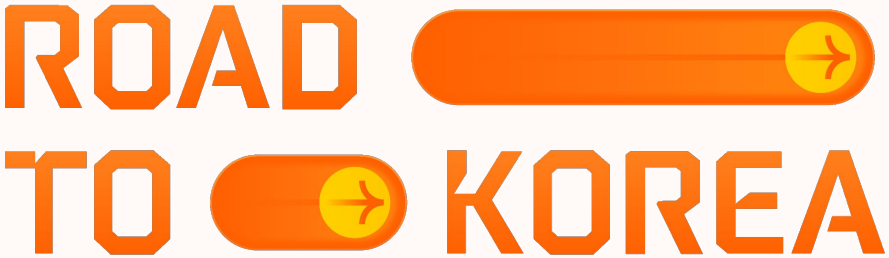
- **@Scope:** Spring Bean Scope quy định phạm vi tồn tại của các bean được Spring container quản lý. Sử dụng **@Scope** annotation để cấu hình phạm vi của bean:
  - Ví dụ: **@Scope("prototype")**

Scope	Description
Singleton	Single Bean per Spring Container - Default
Prototype	New Instance each time Bean is Requested
Request	New Instance per each HTTP Request
Session	New Instance per each HTTP Session
Application	One Instance per each ServletContext
Websocket	One Instance per each WebSocket

Spring Bean, Bean Scope

- **Singleton** (default scope)**:**
  - 1 bean có phạm vi **singleton** chỉ có 1 instance duy nhất trong Spring container. Đây là scope mặc định.
- **Prototype:**
  - 1 bean có phạm vi **prototype** sẽ tạo ra một instance mới mỗi khi được yêu cầu từ Spring container.
- **Request:**
  - 1 bean có phạm vi request sẽ tạo ra một instance mới cho mỗi request HTTP. Phạm vi này chỉ áp dụng trong ngữ cảnh ứng dụng web.
- **Session:**
  - 1 bean có phạm vi session sẽ tạo ra một instance mới cho mỗi session HTTP. Phạm vi này chỉ áp dụng trong ngữ cảnh ứng dụng web.
- **Application:**
  - 1 bean có phạm vi application sẽ tạo ra một instance mới cho mỗi lifecycle của ứng dụng servlet. Phạm vi này chỉ áp dụng trong ngữ cảnh ứng dụng web.
- **WebSocket:**
  - 1 bean có phạm vi websocket sẽ tạo ra một instance mới cho mỗi WebSocket session.

TẠI SAO CÓ @Component RỒI, VẪN CÓ @Bean?



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

mail@mail.com hoặc Zalo 0xxx xxx xxx