Bean Scope
Phạm vi của Bean

Bean Scope

Trong Spring Framework, "Bean Scope" xác định phạm vi sống của một đối tượng (bean) được quản lý bởi Spring IoC (Inversion of Control) container. Bean Scope quyết định một bean được tạo ra như thế nào, lưu trữ và chia sẻ trong ứng dụng.

Spring hỗ trợ các phạm vi bean sau:

- Singleton: Đây là phạm vi mặc định cho các bean trong Spring. Mỗi lần yêu cầu bean, container chỉ trả về một phiên bản duy nhất của bean đó. Nếu bean đã tồn tại, container sẽ cung cấp bean đó, ngược lại container sẽ tạo mới bean và lưu trữ để sử dụng cho các yêu cầu tiếp theo.

Spring hỗ trợ các phạm vi bean sau:

- Prototype: Khi một bean được định nghĩa với phạm vi prototype, mỗi lần yêu cầu bean, container sẽ tạo một phiên bản mới của bean đó. Điều này có nghĩa là mỗi yêu cầu bean đều nhận được một đối tượng độc lập. Đối tượng này không được lưu trữ trong container và không được chia sẻ giữa các yêu cầu.

Spring hỗ trợ các phạm vi bean sau:

- Request: Phạm vi request chỉ áp dụng cho các ứng dụng web. Mỗi lần một yêu cầu HTTP mới được tạo ra, Spring container sẽ tạo ra một phiên bản mới của bean để phục vụ yêu cầu đó. Khi yêu cầu hoàn thành, đối tượng bean được hủy.

Spring hỗ trợ các phạm vi bean sau:

- Session: Phạm vi session cũng áp dụng cho các ứng dụng web. Mỗi phiên làm việc (session) sẽ có một phiên bản duy nhất của bean được tạo ra và chia sẻ trong suốt phiên đó. Khi phiên làm việc kết thúc, bean được hủy.

Spring hỗ trợ các phạm vi bean sau:

- Global Session: Phạm vi global session cũng áp dụng cho các ứng dụng web và tương tự như phạm vi session. Tuy nhiên, nó chỉ áp dụng trong các môi trường phân tán (distributed environment), ví dụ như sử dụng Spring Portlet MVC. Global session bean tồn tại trong suốt quá trình làm việc của toàn bộ ứng dụng web phân tán.

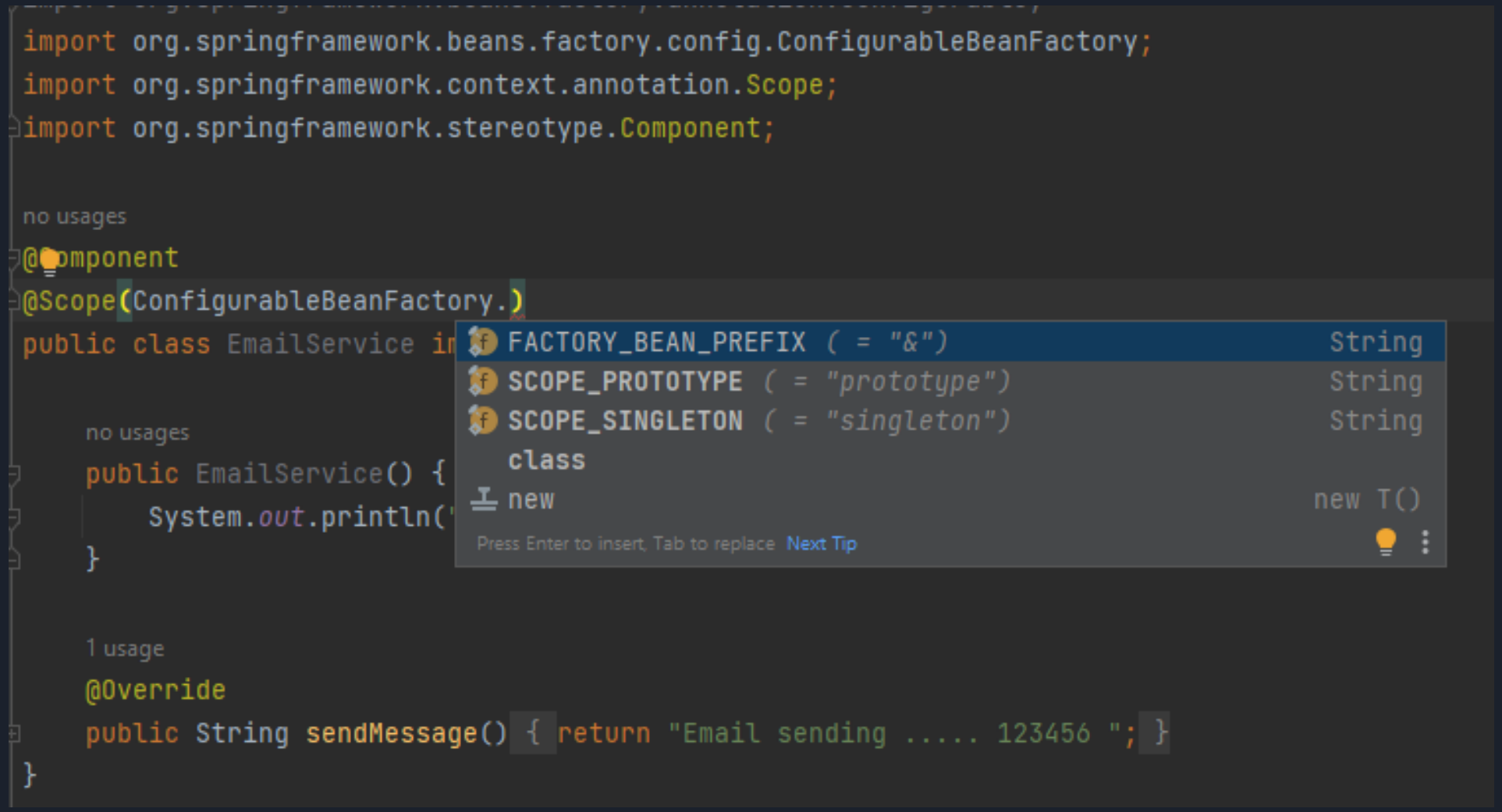
Spring hỗ trợ các phạm vi bean sau:

- Application: Phạm vi application chỉ áp dụng cho các ứng dụng web và tương tự như phạm vi singleton. Bean được tạo ra một lần duy nhất và chia sẻ cho toàn bộ ứng dụng web. Bean này tồn tại cho đến khi ứng dụng web bị dừng.

Spring hỗ trợ các phạm vi bean sau:

- WebSocket: Phạm vi WebSocket áp dụng cho các ứng dụng sử dụng giao thức WebSocket. Mỗi phiên kết nối WebSocket sẽ có một phiên bản duy nhất của bean, và khi phiên kết nối kết thúc, bean được hủy.

Chúng ta có thể quy định Scope cho các Bean được không?



Singleton

```java
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class MyApp {

    private MyInterface myInterface1;
    private MyInterface myInterface2;

    @Autowired
    public MyApp(MyInterface myInterface1, MyInterface myInterface2) {
        this.myInterface1 = myInterface1;
        this.myInterface2 = myInterface2;
    }

    public static void main(String[] args) {
        SpringApplication.run(MyApp.class, args);
    }

    // Các logic khác của ứng dụng
}
```

Prototype

```java
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class MyApp {

    private MyInterface myInterface1;
    private MyInterface myInterface2;

    @Autowired
    public MyApp(MyInterface myInterface1, MyInterface myInterface2) {
        this.myInterface1 = myInterface1;
        this.myInterface2 = myInterface2;
    }

    public static void main(String[] args) {
        SpringApplication.run(MyApp.class, args);
    }

    // Các logic khác của ứng dụng
}
```

Thực hành

- Xây dựng lại ứng dụng nhắn tin
- Cấu hình Bean Scope lần lượt là Singleton và Prototype
- Hiện thực một request dạng get cho biết chương trình đang được sử dụng là ở dạng Singleton hay là Prototype