

1.3

Beans




Beans

A "bean" is a managed object that is instantiated, assembled, and managed by the Spring IoC container.

Beans form the backbone of a Spring application and are the core building blocks that are wired together to create the application.

```
Car obj = new Car();
```



Spring Annotations

- Traditionally, Spring allows a developer to manage bean dependencies by using XML-based configuration.
- There is an alternative way to define beans and their dependencies. This method is a Java-based configuration.
- Unlike the XML approach, Java-based configuration allows you to manage bean components programmatically. That's why Spring annotations were introduced.

Override

Defining Beans

1. Using Stereotype Annotations

Annotate your class with one of the stereotype annotations (`@Component`, `@Service`, `@Repository`, `@Controller`). These annotations inform Spring that the class should be managed as a bean.

2. Explicit Bean Declaration in Configuration Class

Create a configuration class and annotate it with `@Configuration`. This class will contain methods to define and configure beans.

Beans Lifecycle

Bean Created

The bean instance is created by invoking a static factory method or an instance factory method (for annotation-based configuration).

Dependency Injected

Spring sets the bean's properties and dependencies, either through setter injection, constructor injection, or field injection.

Bean Initialized

If the bean implements the `InitializingBean` interface or defines a custom initialization method annotated with `@PostConstruct`, Spring invokes the initialization method after the bean has been configured.

Bean is Used

The bean is now fully initialized and ready to be used by the application.

Bean Destroyed

Spring invokes the destruction method when the bean is no longer needed or when the application context is being shut down.

Bean Lifecycle Hooks

The **@PostConstruct** annotation is used to mark a method that should be invoked immediately after a bean has been constructed and all of its dependencies have been injected.

The **@PreDestroy** annotation is used to mark a method that should be invoked just before a bean is destroyed by the container. This method can perform any necessary cleanup or resource releasing tasks.

Scope of Beans

Scope	Description
singleton	(Default) Scopes a single bean definition to a single object instance for each Spring IoC container.
prototype	Scopes a single bean definition to any number of object instances.
request	Scopes a single bean definition to the lifecycle of a single HTTP request. That is, each HTTP request has its own instance of a bean created off the back of a single bean definition. Only valid in the context of a web-aware Spring ApplicationContext.
websocket	Scopes a single bean definition to the lifecycle of a WebSocket. Only valid in the context of a web-aware Spring ApplicationContext.

