Today's topic : Bean Life Cycle

06 December 2021 21:33

=> The Java class which is managing by IOC is called as Spring Bean

=> To represent java class as spring bean we have several annotations

@Component
@Service
@Repository
@Controller
@RestController

@Configuration
@Bean

=> IOC managing Spring Bean means it will take care of bean object creation and destruction.

=> Bean life cycle means from object creation to object destruction process.

=> For every bean we can execute life cycle methods like init ( ) and destory ( )

=> If we want to perform some operations when bean object got created and when bean object getting removed then we can go bean life cycle methods

=> We can work with bean life cycle methods in 3 ways

1) Programmatic approach
2) Annotation approach
3) XML Approach (Not supported in Spring Boot)

==========

=> To work with bean life cycle programmatically we have to implement below 2 interfaces

1) InitializingBean ===> afterPropertiesSet( )
2) DisposableBean ===> destroy ( )

=> afterPropertiesSet( ) method will be called when bean object got created
=> destroy( ) method will be called before removing bean object

---

```
@Component
public class Motor implements InitializingBean, DisposableBean {

	public Motor() {
		System.out.println("*** Motor :: Constructor ***");
	}

	@Override
	public void afterPropertiesSet() throws Exception {
		System.out.println(" **afterPropertiesSet() **");
	}

	@Override
	public void destroy() throws Exception {
		System.out.println("** destroy( ) **");
	}
}
```

=> To work with Annotation Based Bean Life Cycle we will use below 2 annotations

1) @PostConstruct
2) @PreDestory

```
@Component
public class Engine {

	public Engine() {
		System.out.println("** Engine :: Constructor **");
	}

	@PostConstruct
	public void init() {
		System.out.println(" ** Engine :: init( ) method **");
	}

	@PreDestroy
	public void destroy() {
		System.out.println(" ** Engine :: destroy( ) method **");
	}
}
```

=> @PostConstruct annotated method will execute after bean object got created

=> @PreDestroy annotated method will execute before bean object removal

==========

1) What is Spring Framework
2) Spring Framework Architecture
3) Spring Modules
4) IOC Container
5) Dependency Injection
6) Developing First Spring Application with XML Approach
7) Setter Injection with Example (<property/> tag)
8) Constructor Injection with Example (<constructor-arg />)
9) Bean Scopes (singleton, prototype)
10) What is Spring Boot
11) Advantages of Spring Boot
    a) Starter Poms
    b) Version Conflicts Solution
    c) Auto Configuration
    d) Embedded Servers
    e) Actuators

12) Spring Boot Application Creation
    1) Start.spring.io
    2) STS

13) Spring Boot Application Folder Structure
14) Spring Boot Start Class
15) SpringApplication.run ( ) method
16) @SpringBootApplication annotation
17) Component Scanning
18) Base Package Naming Convention
19) Stereotype Annotations
    1) @Component
    2) @Service
    3) @Repository
    4) @Controller
    5) @RestController
20) @Configuration
21) @Bean

22) @Autowiring
23) Autowiring with Setter, Constructor & Field

24) How to access private variables using Reflection
25) Internals of field injection
26) Spring Boot Banner
27) Runners in Spring Boot
28) Bean Life Cycle