

Spring Boot Startup Hooks

ApplicationContextInitializer vs CommandLineRunner

Slide 1: Title

Spring Boot Startup Hooks

ApplicationContextInitializer vs CommandLineRunner

Slide 2: Startup Timeline (Big Picture)

```
JVM starts
↓
static blocks
↓
main()
↓
SpringApplication.run()
↓
ApplicationContextInitializer ← HERE
↓
Bean definitions loaded
↓
Beans instantiated
↓
ApplicationContext refreshed
↓
CommandLineRunner / ApplicationRunner
```

Slide 3: ApplicationContextInitializer – What is it?

- Spring interface to **customize ApplicationContext before refresh**
- Runs **before beans are created**
- Used for **environment-level setup**

```
public class ProfileInitializer
    implements
    ApplicationContextInitializer<ConfigurableApplicationContext> {

    @Override
    public void initialize(ConfigurableApplicationContext context) {
        context.getEnvironment().setActiveProfiles("dev");
    }
}
```

```
}  
}
```

Slide 4: ApplicationContextInitializer – Use Cases

- Set active profiles programmatically
- Load external properties
- Modify environment variables
- Register infrastructure beans
- Framework / starter initialization

Slide 5: ApplicationContextInitializer – Key Points

- Runs **before** IOC container refresh
- Beans are **NOT** available
- Cannot use @Autowired
- Ideal for **configuration & setup** logic

Slide 6: CommandLineRunner – What is it?

- Spring Boot interface to run code **after** application startup
- Runs **after** beans are created
- Used for **post-startup** logic

```
@Component  
public class MyRunner implements CommandLineRunner {  
  
    @Override  
    public void run(String... args) {  
        System.out.println("Application started");  
    }  
}
```



Slide 7: CommandLineRunner – Use Cases

- Data initialization
 - Calling services/repositories
 - Startup validation
 - Running batch jobs
 - Logging startup info
-

Slide 8: CommandLineRunner – Key Points

- Runs **after ApplicationContext is ready**
 - All beans are available
 - Supports @Autowired
 - Business logic allowed
-

Slide 9: Comparison Table

Feature	ApplicationContextInitializer	CommandLineRunner
Execution time	Before beans	After beans
Bean access	 No	 Yes
Main purpose	Context setup	Startup logic
Common usage	Profiles, env	Data load, jobs

Slide 10: When to Use Which?

Use ApplicationContextInitializer when: - You need to set profiles - You need to modify environment - Beans should be affected by config

Use CommandLineRunner when: - You need beans - You want to execute logic after startup - You are initializing data

Slide 11: Interview One-Liners

- *ApplicationContextInitializer runs before the Spring context is refreshed.*
 - *CommandLineRunner runs after the application is fully started.*
-

Slide 12: Summary

- ApplicationContextInitializer → **prepare context**
- CommandLineRunner → **run logic after startup**
- Both serve different phases of Spring Boot lifecycle