Spring Boot Application Start in Deep

START HERE

Start Application @SpringBootApplication with main()

1
1. Start Application (@SpringBootApplication with main())

The **@SpringBootApplication** annotation marks the main class of your Spring Boot application.
It combines three crucial annotations: **@Configuration**, **@EnableAutoConfiguration**, and **@ComponentScan**, serving as the entry point of the application.

Create Spring Application Instance

2
2. Create SpringApplication Instance

SpringApplication class is instantiated to bootstrap your Spring Boot application. This class sets up the initial configuration, determines the type of application (e.g., web or non-web), processes command-line arguments, and prepares the application to run.

Prepare Environment

3
3. Prepare Environment

Spring Boot sets up the application environment before creating the application context. It involves loading property sources, environment variables, system properties, and command-line arguments.

Load Application Properties

4
4. Load Application Properties

Spring Boot loads configuration properties that determine the application's behavior. These properties are typically defined in **application.properties** or **application.yml** files located in the **src/main/resources** directory or in external locations.

Configure Logging

5
5. Configure Logging

Spring Boot initializes the logging system to capture log messages from the application and the Spring framework itself. By default, Spring Boot uses **Logback** as the logging framework and provides sensible defaults that can be customized.

Create Application Context

6
6. Create Application Context

Spring Boot creates the **ApplicationContext**, which is the central container that manages the beans and configurations for your application.
The **ApplicationContext** is responsible for:
- **Bean Instantiation and Management.**
- **Dependency Injection.**
- **Resource Loading.**

Register Application Listeners

7
7. Register Application Listeners

Spring Boot registers **Application Listeners** that are used to listen for application events during the startup process and throughout the application's lifecycle. **Application listeners** are components that respond to events published by the application context, such as context refreshes, application starting or stopping, and custom events defined by your application.

Initialize Banner (Display Spring Boot Banner)

8
8. Initialize Banner (Display Spring Boot Banner)

Spring Boot initializes and displays the application banner in the terminal or console. By default, this banner includes the Spring Boot version and a simple ASCII art logo. The banner is displayed just before the application context is refreshed, serving as a visual indication that the application is starting up.

Trigger Refresh ApplicationContext

9
9. Trigger Refresh ApplicationContext

Spring Boot refreshes the **ApplicationContext**, completing the initialization of the application context and making all beans ready for use. The refresh process involves several key actions

Scan Components and Beans

10
10. Scan Components and Beans

Spring Boot refreshes the **ApplicationContext**, completing the initialization of the application context and making all beans ready for use. The refresh process involves several key actions.

Load Bean Definitions

11
11. Load Bean Definitions

Spring Boot loads the bean definitions into the application context. **Bean** definitions are metadata that tell the Spring container how to instantiate, configure, and assemble the objects (beans) in your application.

Perform Auto-Configuration

12
12. Perform Auto-Configuration

Spring Boot performs auto-configuration by applying its **auto-configuration** classes to the application context. **Auto-configuration** is a core feature of Spring Boot that automatically configures your application based on the dependencies present on the classpath and defined properties.

Instantiate Beans (Dependency Injection)

13
13. Instantiate Beans (Dependency Injection)

Spring Boot instantiates the beans defined in the application context and performs dependency injection to assemble the application's components

Apply Post Processors

14
14. Apply Post Processors

Spring Boot instantiates the beans defined in the application context and performs dependency injection to assemble the application's componentsSpring Boot applies post-processors to the beans within the application context. Post-processors are mechanisms provided by the Spring framework that allow for custom modification of bean definitions and instances during the container's startup process. There are two primary types of post-processors.

Publish Application Events

15
15. Publish Application Events

Spring Boot publishes application events to notify interested components about specific stages in the application lifecycle. These events allow different parts of the application to respond to context changes, application readiness, and other significant occurrences without tight coupling.

Prepare CommandLineRunners

16
16. Prepare CommandLineRunners

Spring Boot identifies and prepares beans that implement the **CommandLineRunner** or **ApplicationRunner** interfaces. These special beans contain code that you want to execute after the application context is fully initialized but before the application is considered ready.

Run CommandLineRunners and ApplicationRunners

17
17. Run CommandLineRunners and ApplicationRunners

In this step, after the application context has been fully initialized and all beans have been instantiated and post-processed, Spring Boot executes all beans that implement the **CommandLineRunner** and **ApplicationRunner** interfaces. This allows you to run specific code immediately after the application has started but before it is considered fully ready to accept requests (especially in web applications).

Application Started and Ready

18
18. Application Started and Ready

At this final step, the **Spring Boot** application is fully started and ready to handle requests or perform its designated tasks.

http://

AVAILABLE

The Dev Project