Presents

Spring Boot

Learning Objectives

- In this chapter we will:
  - ✓ Learn what a Spring Container is and its role
  - ✓ Learn how to create and manage Spring Beans
  - ✓ Learn how Spring implements IoC
  - ✓ Learn now Spring manages dependencies using DI
  - ✓ Learn about Bean lifecycles

Complexity

- Spring simplifies the organization of POJOs into a complex Java application
- However, this does not make the application simple to deploy and configure
- We still have to manage:
  - ✓ Writing and coding all the configuration metadata
  - ✓ Configuring the toolchain – Maven, etc
  - ✓ Configuring the application's integration with other components
    - Databases, files, sockets
    - Web compontents

Convention over Configuration

- Spring framework manages the POJOs in the application architecture
  - ✓ However, the configuration of Spring can be very complex
  - ✓ Most of the configurations for a project are often similar
  - ✓ Similar applications often use similar architecture
- "Convention over Configuration"
  - ✓ A labor-saving approach
  - ✓ Rather than build every configuration from scratch we just follow convention and do what everyone else does
  - ✓ This is called an "opinionated" approach because had definite opinions about what ought to be done
  - ✓ We trade flexibility in choices for ease of development

Spring Boot

- Spring Boot is an application framework that uses an opinioned set of defaults to simplify configuration and deployment of a Java application
- Opinionated means
  - ✓ The defaults used by Spring Boot are reasonable
  - ✓ But you can override them in the configuration
- For example:
  - ✓ You can use any web container in a Boot app
  - ✓ But defaulting to Tomcat is a reasonable convention
- There are many preconfigured shortcuts like using Spring Boot starters

Starters

- A starter is a set of dependencies specific to a type of application
  - ✓ A list of starters is available at:

    https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/#using.build-systems.starters
- For example,
  - ✓ `spring-boot-starter-web`
  - ✓ "Starter for building web, including RESTful, applications using Spring MVC. Uses Tomcat as the default embedded container"
- Spring Boot apps can be totally self contained
  - ✓ A single jar file that contains everything the app needs to run
  - ✓ Including a web server if necessary
  - ✓ Can also be deployed as a WAR file

Spring Initializr

- The Spring Initialize is located at:
  - ✓ https://start.spring.io
- It allows you to quickly configure a Spring boot application
  - ✓ The Spring projects or components you want are selected from a list
  - ✓ Spring boot autowires all of them together into a project
- The resulting deliverable is a Maven or Gradle project that can be built
  - ✓ With either a WAR file packaging for delivery to an existing server
  - ✓ Or a standalone JAR file that has the server in it.

Summary

- This module has introduced the basic concepts in the Spring core framework
  - ✓ Spring uses containers to implement IoC
  - ✓ Manages and coordinates Java objects
  - ✓ Resolves and implements dependencies
  - ✓ Manages the lifecycles of the Java objects
- There is a lot more to the Spring framework
  - ✓ Discussion in depth or of other modules and projects in the Spring platform is beyond the scope this first introduction

Questions