

# 学习目的与要求

本章首先介绍什么是 Spring Boot,然后介绍 Spring Boot 应用的开发环境,最后介绍如何快速地构建一个 Spring Boot 应用。通过本章的学习,读者应该掌握如何构建 Spring Boot 应用的开发环境以及 Spring Boot 应用。

## 本章主要内容

- Spring Boot 概述
- 快速地构建 Spring Boot 应用

从前两章的学习可知,Spring 框架非常优秀,问题在于"配置过多",造成开发效率低、部署流程复杂以及集成难度大等问题。为了解决上述问题,Spring Boot 应运而生。在编写本书时, Spring Boot 的最新正式版本是 3.0.2,读者在测试本书示例代码时建议使用 3.0.2 或更高版本。

# 3.1 Spring Boot 概述

# ▶ 3.1.1 什么是 Spring Boot

Spring Boot 是由 Pivotal 团队提供的全新框架,其设计目的是简化新 Spring 应用的初始 搭建以及开发过程。使用 Spring Boot 框架可以做到专注于 Spring 应用的开发,无须过多关 注样板化的配置。

在 Spring Boot 框架中,使用"约定优于配置(Convention Over Configuration,COC)"的理 念。针对企业应用开发,Spring Boot 提供了符合各种场景的 spring-boot-starter 自动配置依 赖模块,这些模块都是基于"开箱即用"的原则,进而使企业应用开发更加快捷和高效。可以 说,Spring Boot 是开发者和 Spring 框架的中间层,目的是帮助开发者管理应用的配置,提供 应用开发中常见配置的默认处理(即约定优于配置),简化 Spring 应用的开发和运维,降低开 发者对框架的关注度,使开发者把更多精力放在业务逻辑代码上。通过"约定优于配置"的原 则,Spring Boot 致力于在蓬勃发展的快速应用开发领域成为领导者。

# ▶ 3.1.2 Spring Boot 的优点

Spring Boot 之所以能够应运而生,是因为它具有以下优点。

- (1) 使编码变得简单: 推荐使用注解。
- (2) 使配置变得快捷:具有自动配置、快速构建项目、快速集成第三方技术的能力。
- (3) 使部署变得简便:内嵌 Tomcat、Jetty 等 Web 容器。
- (4) 使监控变得容易:自带项目监控。

# ▶ 3.1.3 Spring Boot 的主要特性

Spring Boot 的主要特性如下。

## ● 约定优于配置

Spring Boot 遵循"约定优于配置"的原则,只需要很少的配置,大多数情况直接使用默认

配置即可。

#### 2 独立运行的 Spring 应用

Spring Boot 可以以 JAR 包的形式独立运行。使用 java - jar 命令或者在项目的主程序中 执行 main()方法运行 Spring Boot 应用(项目)。

#### 3 内嵌 Web 容器

Spring Boot 可以选择内嵌 Tomcat、Jetty 等 Web 容器,无须以 WAR 包形式部署应用。

#### ④ 提供 starter 简化 Maven 配置

Spring Boot 提供了一系列的 starter pom 简化 Maven 的依赖加载,基本上可以做到自动 化配置、高度封装、开箱即用。

#### **6** 自动配置 Spring

Spring Boot 根据项目依赖(在类路径中的 JAR 包、类)自动配置 Spring 框架,极大地减少 了项目的配置。

#### ● 提供准生产的应用监控

Spring Boot 提供了基于 HTTP、SSH、Telnet 对运行的项目进行跟踪监控。

#### ⑦ 无代码生成和 XML 配置

Spring Boot 不是借助于代码生成实现的,而是通过条件注解实现的,建议使用 Java 配置 和注解配置相结合的配置方式,这样方便、快捷。

# 3.2 第一个 Spring Boot 应用

# ▶3.2.1 Maven 简介

Apache Maven 是一个软件项目管理工具。基于项目对象模型(Project Object Model, POM)的理念,通过一段核心描述信息管理项目构建、报告和文档信息。在 Java 项目中, Maven 主要完成两项工作:①统一开发规范与工具;②统一管理 JAR 包。

Maven 统一管理项目开发所需要的 JAR 包,这些 JAR 包不再包含在项目内(即不在 lib 目录下),而是存放于仓库中。仓库主要包括以下内容。

#### 中央仓库

中央仓库存放开发过程中的所有 JAR 包,例如 JUnit,可以通过互联网下载,下载地址为 http://mvnrepository.com。

#### 2 本地仓库

本地仓库指本地计算机中的仓库。下载 Maven 的本地仓库配置在"% MAVEN\_HOME%\conf\settings.xml"文件中,找到 localRepository 即可。

Maven 项目首先从本地仓库中获取所需要的 JAR 包,当无法获取指定的 JAR 包时,本地 仓库将从远程仓库(中央仓库)中下载 JAR 包,并存入本地仓库以备将来使用。

# ▶ 3.2.2 Maven 的 pom.xml

Maven 是基于项目对象模型的理念管理项目的,所以 Maven 的项目都有一个 pom.xml 配置文件管理项目的依赖以及项目的编译等功能。

在 Maven Web 项目中重点关注以下元素。

## **①** properties 元素

在<properties></properties>之间可以定义变量,以便在<dependency></dependency>

## 中引用,示例代码如下:

```
<properties>
<properties>
<properties>
<properties>
</properties>
<dependencies>
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-core</artifactId>
<version>${spring.version}</version>
</dependency>
</dependency>
</dependency>
</dependency>
</dependency>
</dependency>
</dependencies>
```

#### **2** dependencies 元素

<dependencies> </dependencies > 元素包含多个项目依赖需要使用的 < dependency >
</dependency>元素。

#### **3** dependency 元素

在<dependency></dependency>元素内部通过<groupId></groupId>、<artifactId></artifactId>、<version></version>3个子元素确定唯一的依赖,也可以称为3个坐标。示例 代码如下:

<dependency>

```
<!--groupId 组织的唯一标识-->
<groupId>org.springframework</groupId>
<!--artifactId项目的唯一标识-->
<artifactId>spring-core</artifactId>
<!--version项目的版本号 -->
<version>${spring.version}</version>
</dependency>
```

## **4** scope 子元素

在<dependency></dependency>元素中有时使用<scope></scope>子元素管理依赖的部署。<scope></scope>子元素可以使用以下5个值:

#### 1) compile(编译范围)

compile 是默认值,即默认范围。如果依赖没有提供范围,那么该依赖的范围就是编译范围。编译范围的依赖在所有的 classpath 中可用,同时也会被打包发布。

## 2) provided(已提供范围)

provided 表示已提供范围,只有当 JDK 或者容器已提供该依赖时才可以使用。已提供范围的依赖不是传递性的,也不会被打包发布。

# 3) runtime(运行时范围)

runtime 在运行和测试系统时需要,但在编译时不需要。

4) test(测试范围)

test 在一般的编译和运行时都不需要,它们只有在测试编译和测试运行阶段可用。该依赖不会随项目发布。

## 5) system(系统范围)

system 范围与 provided 范围类似,但需要显式提供包含依赖的 JAR 包, Maven 不会在 Repository 中查找它。

# ▶3.2.3 使用 IntelliJ IDEA 快速构建 Spring Boot 应用

IntelliJ IDEA 简称 IDEA,是 Java 编程语言的集成开发环境。IDEA 被业界公认为较好

的 Java 开发工具,尤其在智能代码助手、代码自动提示、重构、Java EE 支持、各类版本工具 (Git、SVN等)、JUnit、CVS 整合、代码分析、创新的 GUI 设计等方面的功能可以说是超常的。 用户可以登录官网,根据操作系统下载相应的 IntelliJ IDEA。本书使用的是 Windows Ultimate 版 ideaIU-2022.2.1(内置 Java 运行环境 Open JDK)。

下面详细讲解如何使用 IDEA 集成开发工具快速构建一个 Spring Boot Web 应用,其具体实现步骤如下。

## ● 新建 Spring Project

打开 IDEA,选择 File→New→Project 命令,打开 New Project 对话框。在该对话框的左 侧选择 Spring Initializr 选项,打开如图 3.1 所示的界面输入项目信息。

😰 New Project			×
Q New Project			
Empty Project			
Generators			
m Maven Archetype			
🥒 Jakarta EE			
ng Spring Initializr		lava Kotlin Groovy	
📭 JavaFX			
💽 Quarkus		Gradle - Groovy Gradle - Kotlin Maven	
$\mu$ Micronaut			
🍫 Ktor			
Kotlin Multiplatform			
Compose Multiplatform		com ch ch 2	
5 HTML	Fackage hame.		
🏶 React		🗮 18 Oracle OpenJDK version 18.0.2 🔹 🔻	
ex Express			
A Angular CLI	Java:		
IDE Plugin		Jar War	
🛎 Android			
?		Next Cancel	

图 3.1 输入项目信息

## 2 选择项目依赖

在图 3.1 中输入项目信息后,单击 Next 按钮,打开如图 3.2 所示的选择项目依赖界面。 在图 3.2 中选择项目依赖(如 Spring Web)后,单击 Create 按钮,即可完成 Spring Boot Web 应 用的创建。

## 3 编写测试代码

在 ch3 应用的 src/main/java 目录下创建 com.ch.ch3.test 包,并在该包中创建 TestController 类,代码如下:

```
package com.ch.ch3.test;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;
@RestController
public class TestController {
    @GetMapping("/hello")
    public String hello() {
        return "您好,Spring Boot!";
    }
}
```

69



图 3.2 选择项目依赖

在上述代码中使用的@RestController 注解是一个组合注解,相当于 Spring MVC 中的 @Controller和@ResponseBody 注解的组合,具体应用如下:

(1) 如果只是使用@RestController 注解 Controller,则 Controller 中的方法无法返回 JSP、HTML 等视图,返回的内容就是 return 的内容。

(2)如果需要返回到指定页面,则需要使用@Controller 注解。如果需要返回 JSON、 XML或自定义 mediaType 内容到页面,则需要在对应的方法上加上@ResponseBody 注解。

# ④ 生成应用程序的 App 类

在 ch3 应用的 com.ch.ch3 包中自动生成了应用程序的 App 类 Ch3Application。这里省 略 Ch3Application 的代码。

## 5 运行 main 方法启动 Spring Boot 应用

运行 Ch3Application 类的 main 方法, 控制台信息如图 3.3 所示。

Rur	n:	Ch3Application ×						\$
G	Cor							
<b>チ</b> 目 回 載	← → I? ‡	/// /'(_) ( ( )/   '   '    ' / / //   '   '   ' / / //               (. '   !						
∃ ■	÷		(v3.0.2	2)				
*		2023-02-01T06:22:13.076+08:00 2023-02-01T06:22:13.081+08:00 2023-02-01T06:22:14.302+08:00 2023-02-01T06:22:14.315+08:00	INFO 4008 INFO 4008 INFO 4008 INFO 4008	[ [ [	main] main] main] main]	com.ch.ch3.Ch3Application com.ch.ch3.Ch3Application o.s.b.w.embedded.tomcat.TomcatWebServ o.apache.catalina.core.StandardServi	ver :	: Starting Ch3Application u : No active profile set, fa : Tomcat initialized with p : Starting service [Tomcat]
					<b>.</b>	(古田已站的周人后百		

图 3.3 启动 Spring Boot 应用后的控制台信息

从控制台信息可以看到 Tomcat 的启动过程、Spring MVC 的加载过程。

注意: Spring Boot 3.0 内嵌了 Tomcat 10,因此对于 Spring Boot 应用不需要开发者配置与启动 Tomcat。

#### **6** 测试 Spring Boot 应用

在启动 Spring Boot 应用后,默认访问地址为"http://localhost:8080/",将项目路径直接

设为根路径,这是 Spring Boot 的默认设置。因此,可以通过 "http://localhost: 8080/hello"测试应用(hello 与测试类 TestController中的@GetMapping("/hello")对应),测试效果如 图 3.4 访问 Spring Boot 应用 图 3.4 所示。

# ▶ 3.2.4 使用 Spring Tool Suite 快速构建 Spring Boot 应用

Spring Tool Suite(STS)是一个定制版的 Eclipse,专为 Spring 开发定制,方便创建、调试、运行、维护 Spring 应用。通过该工具,可以很轻易地生成一个 Spring 工程,例如 Web 工程,最令人兴奋的是工程中的配置文件都将自动生成,开发者再也不用关注配置文件的格式及各种配置了。用户可以登录 Spring 官网"https://spring.io/tools"下载 Spring Tools for Eclipse,本书使用的版本是 spring-tool-suite-4-4.17.1.RELEASE-e4.26.0-win32.win32.x86\_64.self-extracting.jar(内置 Java 运行环境 Open JDK)。该版本与 Eclipse 一样,无须安装,解压即可使用。

下面详细讲解如何使用 STS 快速构建一个 Spring Boot 应用,其具体实现步骤如下。

#### ● 新建 Spring Starter Project

选择 File→New→Spring Starter Project 命令,打开如图 3.5 所示的 New Spring Starter Project 对话框。

9				– 🗆	×
New Spring St	arter Project				Ċ
Service URL	https://start.spring	.io			~
Name	ch3				
🗹 Use default	location				
Location	C:\Users\chenheng\	Documents\worksp	ace-sprin	g-tool-suit	Browse
Type:	Maven	<ul> <li>Packagir</li> </ul>	ng: J	ar	~
Java Version:	17	<ul> <li>Languag</li> </ul>	le: J	ava	~
Group	com.ch				
Artifact	ch3				
Version	0.0.1-SNAPSHOT				
Description	Demo project for S	pring Boot			
Package	com.ch.ch3				
Working sets					
Add proje	ct to working sets			Nev	v
Working sets	:			<ul> <li>Sele</li> </ul>	ct
0	< Back	Next >	Finish	Ca	ncel

图 3.5 New Spring Starter Project 对话框

#### 2 选择项目依赖

在图 3.5 中输入项目信息后,单击 Next 按钮,打开如图 3.6 所示的 New Spring Starter Project Dependencies 对话框,在其中选择项目依赖,如 Web。

单击图 3.6 中的 Finish 按钮,即可完成 Spring Boot Web 应用的创建。

## 3 编写测试代码

在 ch3 应用的 src/main/java 目录下创建 com.ch.ch3.test 包,并在该包中创建 TestController 类,代码与 3.2.3 节中相同,这里不再赘述。

#### ❹ 生成应用程序的 App 类

在 ch3 应用的 com.ch.ch3 包中自动生成了应用程序的 App 类 Ch3Application。这里省 略 Ch3Application 的代码。



# 5 运行 main 方法启动 Spring Boot 应用

运行 Ch3Application 类的 main 方法,控制台信息如图 3.7 所示。

I Problems @ Javadoc ╚ Declaration ♀Co Ch3Application [Java Application] [pid: 133	onsole × 328]		今 ■ 저 옷   承 점 와 만만   면 U ▼
/ / / _ · _ · _ · _ · _ · _ · _ · _ · · · ·	())))) //// (v3.0.2)		
2023-01-31709:15:04.841+08:00 INF 2023-01-31709:15:04.847+08:00 INF 2023-01-31709:15:04.079408:00 INF 2023-01-31709:15:06.079408:00 INF 2023-01-31709:15:06.092408:00 INF 2023-01-31709:15:06.227+08:00 INF 2023-01-31709:15:06.227+08:00 INF 2023-01-31709:15:06.704408:00 INF 2023-01-31709:15:06.704408:00 INF	0 13328 [ mai 0 13328 [ mai	<pre>n] com.ch.ch3.Ch3Application n] com.ch.ch3.Ch3Application o] o.s.b.w.embedded.tomcat.TomcatWebServer n] o.apache.catalina.core.StandardService n] o.apache.catalina.core.StandardEngine n] o.a.c.c.C.[Tomcat].[Jocalhost].[/] n] w.s.c.ServletWebServerApplicationContext n] o.s.b.w.embedded.tomcat.TomcatWebServer n] com.ch.ch3.Ch3Application</pre>	: Starting Ch3Application using Java 17.0.5 : No active profile set, falling back to 1 of : Tomcat initialized with port(s): 8880 (htt : Starting Service [Tomcat] : Starting Service medded WebApplication : Initializing Spring embedded WebApplication : Root WebApplicationContext: initialization : Tomcat started on port(s): 8880 (http) wit : Started Ch3Application in 2.388 seconds (t)

#### 图 3.7 启动 Spring Boot 应用后的控制台信息

## **⑥** 测试 Spring Boot 应用

在启动 Spring Boot 应用后,可以通过"http://localhost:8080/hello"测试应用。

# 3.3 本章小结

本章首先简单介绍了 Spring Boot 应运而生的缘由,然后演示了如何使用 IDEA 和 STS 快速构建 Spring Boot 应用。

IDEA 在业界被公认为较好的 Java 开发工具,本书后续章节都使用 IDEA 编写示例代码。 如果开发者要构建 Spring Boot 应用,可以根据实际工程需要选择合适的 IDE。





- 1. Spring、Spring MVC、Spring Boot 三者有什么联系?为什么要学习 Spring Boot?
- 2. 在 IDEA 中如何快速构建 Spring Boot 的 Web 应用?