```
    public static void main(String[] args) {
        SpringApplication.run(MyApplication.class, args);
    }
}
```

以上示例中最重要的部分是运行 IoC 容器时需要一个名为`@SpringBootApplication` 的注解。这里也有 MVC 服务器以及其他应用程序组件。让我们深入研究一下。首先，Spring Boot 是一系列模块，是 Gradle 或 Maven 等现代构建工具的补充。通常，Spring Boot 依赖于两个核心模块。第一个是 `spring-boot` 模块，它带有与 Spring IoC 容器相关的所有可能的默认配置。第二个是 `spring-boot-autoconfigure`，它为所有现有的 Spring 项目（例如 Spring Data、Spring MVC、Spring WebFlux 等）带来了所有可能的配置。乍一看，即使并非必须，但似乎所有已定义的配置都能立即启用。然而，情况并非如此：在引入特定依赖项之前，所有配置都是被禁用的。Spring Boot 为模块定义了一个新概念，这些模块的名称中通常包含单词“`-starter-`”。默认情况下，启动程序不包含任何 Java 代码，但会在 `spring-boot-autoconfigure` 中引入所有相关的依赖项以激活特定配置。有了 Spring Boot，我们就有了`-starter-web` 和`-starter-data-jpa` 模块，这些模块能配置所有必需的基础设施组件，而无须额外的工作。与 Spring Roo 项目相比，Spring Boot 明显更具灵活性。除了可以轻松扩展的默认配置，Spring Boot 还提供了一个流式 API，这使我们可以构建自己的启动器。此 API 能替换默认配置，并让我们自己配置特定的模块。

限于自身目的，本书不会介绍 Spring Boot 的详细信息。但是，Greg L. Turnquist 所著的 *Preview Online Code Files Learning Spring Boot 2.0* 第 2 版非常详细地介绍了 Spring Boot。

5.2 Spring Boot 2.0 中的响应式

由于本书是关于响应式编程的，因此我们不会详细介绍 Spring Boot。但是，正如前一节所述，快速启动应用程序的能力是成功框架的关键要素，因此我们需要弄清楚响应式是如何在 Spring 生态系统中反映出来的。由于 Spring MVC 和 Spring Data 模块的阻塞特性，仅将编程范例改为响应式编程不能使我们获益。因此，Spring 团队决定改变这些模块中的整个范例。为此，Spring 生态系统提供了一系列响应式模块。本节将简要介绍这些模块，而其中大部分模块将在本书后面的章节中专门介绍。

5.2.1 Spring Core 中的响应式

Spring 生态系统的核心模块是 Spring Core 模块。Spring 5.*x* 引入的一个值得注意的增强功能是对响应式流和响应式库的原生支持，其中，响应式库包含 RxJava 1/2 和 Project Reactor 3。

1. 响应式类型转换支持

为了支持响应式流规范所进行的最全面的改进之一是引入了 `ReactiveAdapter` 和 `Reactive-`