**Spring Boot 是什么？**

对于传统的 Spring 开发者来说，即使开发一个简单的功能，也需要进行大量的配置声明，以及大量繁琐的注解。

为了化繁为简，Spring Boot 主要解决了以下问题：

1. Spring Boot 提供极其快速和简化的操作，让 Spring 开发者快速上手。
2. 提供了 Spring 运行的默认配置。（约定优于配置）。
3. 为通用 Spring 项目提供了很多非功能性特性，例如：嵌入式 Server，Security，统计，健康检查，外部配置等等。

Spring Boot 包含几个部分，Auto-configuration、Starters、Cli、Actuator 等等。本文主要介绍 Spring Boot 的常用注解，加载原理，环境变量配置以及 Actuator。

**Spring Boot 注解**

**@SpringBootApplication** 注解是 Spring Boot 开发者遇到的第一个注解。由于很多 Spring 的开发者经常在 **Main 方法**里声明 @Configuration, @EnableAutoConfiguration and @ComponentScan. 所以 Sping Boot 提供了这个便捷的注解: @SpringBootApplication，它的作用等同于前面三个注解，它会加载一些列默认的类，也会默认打开 Spring MVC 的注解（@EnableWebMvc）。

@ConditionOnClass(A.class) 表示在某种调节下，这个@Configuration 才会被加载，这里的条件是 A.class 在 ClassPath 之上。

**Spring Boot 启动流程**

* ApplicationStartedEvent 被触发。
* 日志服务被初始化。
* ApplicationEnvironmentPreparedEvent 被触发。



* 读取 Config 文件。你可以为 Spring 项目配置多个 Profile，例如为开发环境配置 Application-dev.properties，此时Spring 会读取你指定的 Profile 进行加载。
* 调用 EnvironmentPostProcessor 里的内容。
* 日志初始化完成。
* ApplicationPreparedEvent 被触发。
* ApplicationContext 被刷新。
* ContextRefreshedEvent 被触发。
* 嵌入式 Server 连接池开启，接受 HTTP 请求。在启动 Server 之前必须保证 Application Context 的内容处于最新的状态。
* EmbeddedServletContainerInitializedEvent 事件被触发。
* ApplicationReadyEvent 事件被触发。

如果你想捕获这些事情，可以通过实现这些事件的监听器（EventListener），来捕获这些事件进行处理，所以这一整套 Spring Boot 的启动流程是可以扩展定制化的。

**Spring Boot 配置环境变量适应不同运行环境**

应用需要被部署到不同的环境进行测试和发布，环境的配置不同，怎么办？一个 Cloud Native 的应用需要具备适应不同环境的能力，也就是说，代码和配置必须分离，**所有与环境相关的变量应该在程序运行时被注入**，而不是打包在 war/jar 包里。

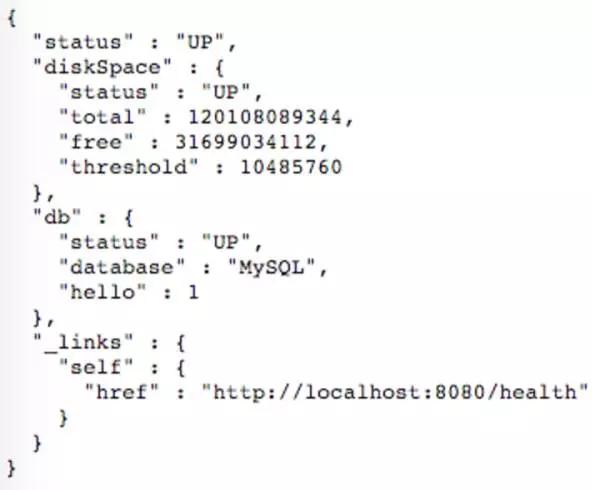
**Spring boot 提供了 EnvironmentPostProcessor 接口来支持你为 Spring 应用注入不同的环境变量。你可以通过实现这个接口，来指定环境变量注入的来源，例如文件，或者云环境里的变量，例如获取 Cloud Foundry 应用运行时环境变量里的数据库连接池信息，ZooKeeper 地址等等。**

**Spring Boot Actuator**

在 Pom 文件中添加 spring-boot-starter-actuator 依赖即可使用 Actuator：



Actuator 为我们提供了非常丰富的 Endpoint，包括：/env、/info、/metrics、/health 等。通过这些 EndPoint，我们可以快速浏览应用的环境变量，健康状态，等等。



不仅如此，Actuator 查看运行时使用的环境变量时还会它会将环境变量的敏感值做脱敏处理，默认任何包含 TOKEN 和 Password 的变量都会显示为\*号，你可以自行声明一些敏感字段。这些功能对于生产环境的问题排除，故障预警都非常有帮助。

**总结**

Spring Boot 并不是要成为 Spring IO 平台里面众多基础项目的替代者，而是**为平台带来另一种开发体验**，从而简化对这些已有技术的使用。对于已经熟悉 Spring 生态系统的开发人员来说，Spring Boot 是一个很理想的选择，对于采用 Spring 技术的新人来说，Spring Boot 提供一种更简洁的方式来使用这些技术。