《SpringBoot实战第4版》学习笔记

Spring 1.0：依赖注入 声明式事物

Spring 2.0：自定义XML命名空间

Spring 2.5：

面向注解的依赖注入模型（@Component和@Autowired注解），以及面向注解的Spring MVC编程模型

Spring 3.0：一套基于Java的全新配置，能够取代XML

Spring 3.1：

一系列以@Enable开头的注解进一步完善这一特性，可以写出一个没有任何XML配置的Spring应用程序

Spring 4.0：

对条件化配置提供了支持，根据应用程序的Classpath、环境和其他因素，运行时决策将决定使用哪些配置，忽略哪些配置。

Spring Boot：

自动配置、起步依赖（starter dependency）、Spring Boot CLI、Actuator（一窥应用程序运行时的内部工作细节，看看Spring应用程序上下文都有哪些Bean，Spring MVC控制器是怎么与路径映射的，应用程序都能取到哪些配置属性等）

Spring Boot四大核心：

①自动配置：针对很多Spring应用程序常见的应用功能，Spring Boot能提供自动提供相关配置。

Classpath

自动配置涉及了：Java持久化API（JPA）、Thymeleaf模板、安全和Spring MVC

②起步依赖：告诉Spring Boot需要什么功能，它就能引入需要的库

Spring Boot的起步依赖基本都以Spring-boot-starter打头，随后是直接代表其功能的名字，比如web、test

Spring Boot的Web起步依赖：

org.springframework.boot:spring-boot-starter-web

③命令行界面：这是Spring Boot的可选特性，借此你只需写代码就能完成完整的应用程序，无需传统项目构建。CLI能检测到你使用了哪些类，它知道要向Classpath中添加哪些起步依赖才能让它运转起来。一旦那些依赖出现在Classpath中，一系列自动配置就会接踵而来，确保启用DispatcherServlet和Spring MVC，这样控制器就能响应HTTP请求了。

④Actuator：让你能够深入运行中的Spring Boot应用程序，一探究竟。Actuator提供了运行时检视应用程序内部情况的能力

包括：Spring应用程序上下文里配置的Bean

Spring Boot的自动配置做的决策

应用程序取到的环境变量、系统属性、配置属性和命令行参数

应用程序里线程的当前状态

应用程序最近处理过的HTTP请求的追踪情况

各种和内存用量、垃圾回收、Web请求以及数据源用量相关的指标

Actuator通过Web端点和shell界面向外界提供信息。

CLI两种下载方式：

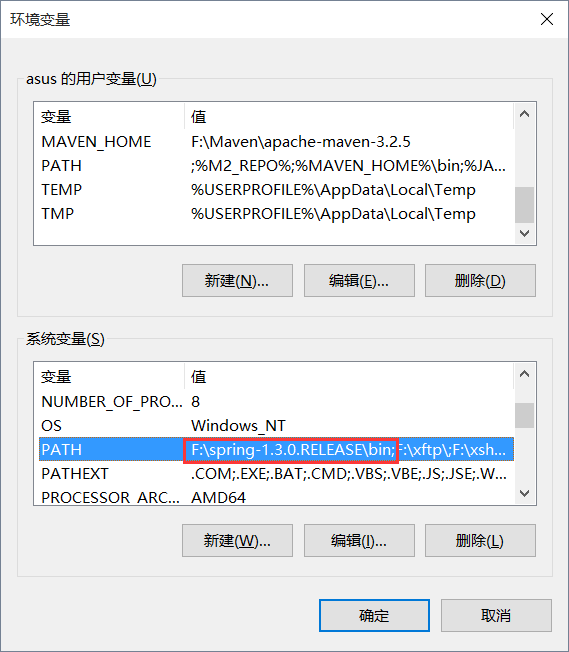
（1）

<http://repo.spring.io/release/org/springframework/boot/spring-boot-cli/1.3.0.RELEASE/spring-boot-cli-1.3.0.RELEASE-bin.zip>

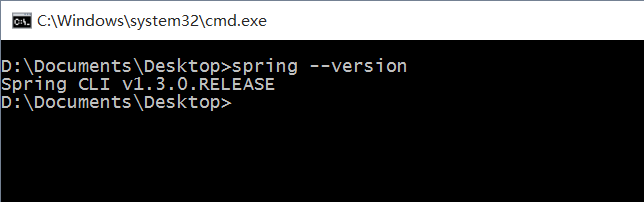
（2）

<http://repo.spring.io/release/org/springframework/boot/spring-boot-cli/1.3.0.RELEASE/spring-boot-cli-1.3.0.RELEASE-bin.tar.gz>

解压后将其包括bin目录的完整路径添加到系统路径中，保存即可。



检验：



开启命令行补全：

注：window上开发，或者没有BASH或zsh，则无法使用自带的命令行补全脚本，但是如果用的是Spring Boot CLI的shell，也有命令补全：

可以使用spring shell 命令开启一个特别针对Spring Boot的shell进程，在里面可以执行各种CLI命令，Tab键也能有命令补全。

Spring Initializr初始化Spring Boot项目：传统方式创建Java项目结构，但是要放弃Groovy语言灵活性的特性，比如自动依赖和import解析

Spring Initializr的几种用法