# 1、Spring Boot提供了哪些核心功能？

1、jar包方式运行

通过引入spring-boot-maven-plugin插件可以将springboot项目打包成一个可以直接运行的jar包，运行方式和常规jar包一样java -jar xxx.jar，启动后可以直接运行内嵌的web容器，根据具体引入的依赖来确定到底该启动哪种web容器。

2、使用了starter依赖

使用starter来封装依赖，简化项目引入相关依赖的复杂度

3、自动配置

springboot将spring4中的 @Condition注解发扬光大，根据特定的条件来创建相关的bean（如classpath下存在某个或者是某些类时自动创建某些spring bean），自动完成相关框架的自动配置。

# 2、你如何理解Spring Boot中的starter？

starter可以理解为启动器，它包含了一系列可以集成到应用里面的依赖包，你可以一站式集成 Spring及其他技术，而不需要到处找示例代码和依赖包。如你想使用spring data redis访问Redis，只要加入spring-boot-starter-data-redis 启动器依赖就能使用了。starter包含了许多项目中需要用到的依赖，它们能快速持续的运行，都是一系列得到支持的管理传递性依赖。

# 3、Spring Boot常用的starter有哪些？

1、spring-boot-starter-web (嵌入tomcat和web开发需要servlet与jsp支持)

2、spring-boot-starter-data-elasticsearch (es支持)

3、spring-boot-starter-data-redis (redis支持)

4、spring-boot-starter-amqp(消息队列支持)

5、spring-boot-starter-data-mongodb (mongodb的支持)

6、mybatis-plus-boot-starter(mybatis plus的支持)

7、mybatis-spring-boot-starter(mybatis的支持)

# **4、Spring Boot的配置⽂件有哪几种格式？**

Spring Boot支持两种格式的配置文件：

1、application.properties

2、application.yml

在实际的项目开发一般的都是使用配置中心管理项目中的配置信息，常见的配置中心：Spring Cloud Config、Nacos

# 5、如何自定义Spring Boot应用程序的端口号？

方式一：在application.properties或者application.yml文件中添加server.port配置项指定端口

方式二：在启动spring boot项目的时候通过-Dserver.port参数指定项目的端口号

方式三：通过WebServerFactoryCustomizer设置端口号，如下所示



# 6、Spring Boot如何定义多套不同环境配置？

提供多套配置文件，如：

applcation.properties

application-dev.properties

application-test.properties

application-prod.properties

然后在applcation.properties文件中指定当前的环境spring.profiles.active=test,这时候读取的就是application-test.properties文件。

# 7、Spring Boot有哪几种读取配置的方式？

Spring Boot 可以通过

1、@Value

2、Environment接口

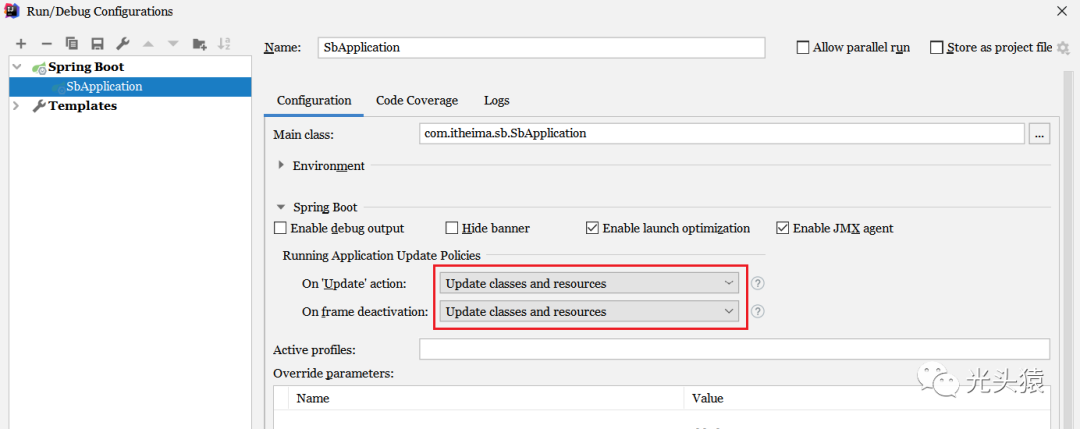
3、@ConfigurationProperties

来绑定变量

# 8、如何重新加载Spring Boot上的更改，而无需重新启动服务器？

使用Spring Boot所提供的devtools工具就可以实现无需重新启动服务器而加载最新的代码。对应的工具的坐标如下所示：

如果使用的是idea开发工具，还需要做如下配置：



由于idea不会自动编译项目，所以源码修改不能被spring-boot-devtools检测到，勾选上面两个选项后idea将在窗口失去焦点时自动编译并替换源码文件。

# 9、Spring Boot的核心注解是哪个？它主要由哪几个注解组成的？

启动类上面的注解是@SpringBootApplication，它也是Spring Boot的核心注解，主要组合包含了以下3个注解：

1、@SpringBootConfiguration：

组合了 @Configuration 注解，实现配置文件的功能。

2、@EnableAutoConfiguration：

打开自动配置的功能，也可以关闭某个自动配置的选项。

如关闭数据源自动配置功能：

@SpringBootApplication(exclude={ DataSourceAutoConfiguration.class })。

3、@ComponentScan：Spring组件扫描。

# 10、运行Spring Boot有哪几种方式？

方式一：直接通过java -jar xxx.jar的方式运行

方式二：将xxx.jar制作成Docker镜像，然后借助于Docker容器进行运行，并且可以使用docker-compose对多个容器进行统一编排

# 11、Spring Boot打成的jar和普通的jar有什么区别？

区别：

1、Spring Boot 项目最终打包成的jar是可执行jar，这种jar可以直接通过 java -jar xxx.jar 命令来运行，这种jar不可以作为普通的 jar 被其他项目依赖，即使依赖了也无法使用其中的类。

2、Spring Boot 的jar无法被其他项目依赖，主要还是他和普通jar的结构不同。普通的jar包，解压后直接就是包名，包里就是我们的代码，而Spring Boot打包成的可执行jar解压后，在 \BOOT-INF\classes目录下才是我们的代码，因此无法被直接引用。

# 12、Spring Boot中如何实现定时任务？

在Spring Boot中可以使用两种定时任务框架：

1、Spring Task

2、Quartz

一般在项目中使用Spring Task就可以了，因为Spring Task是Spring框架提供的可以和Spring Boot进行无缝集成。具体的使用方式如下所示：

1、在启动类上使用@EnableScheduling注解开启定时任务支持

2、在指定的方法上使用@Scheduled注解来指定定时任务的执行规则， 如下所示：

图片

# 13、怎么禁用某些自动配置特性？

如果我们想禁用某些自动配置特性，可以使用 @SpringBootApplication注解的exclude属性来指明。例如，下面的代码段是使

DataSourceAutoConfiguration无效：

图片

# 14、Spring Boot项目的自动化配置原理是什么？

在Spring Boot项目中有一个注解@SpringBootApplication，这个注解是对三个注解进行了封装：

@SpringBootConfiguration、@EnableAutoConfiguration

@ComponentScan

其中@EnableAutoConfiguration是实现自动化配置的核心注解。

该注解的源码如下所示：

图片

该注解通过@Import注解导入AutoConfigurationImportSelector，这个类实现了一个导入器接口ImportSelector。在该接口中存在一个方法selectImports，如下所示：

图片

该方法的返回值是一个数组，数组中存储的就是要被导入到spring容器中的类的全类名。在AutoConfigurationImportSelector类中重写了这个方法，该方法内部就是读取了项目的classpath路径下META-INF/spring.factories文件中的所配置的类的全类名。

如下所示：

