架构师面试题 - Spring Boot 面试专题

目录

1、什么是 spring boot

1. 用来简化 spring 应用的初始搭建以及开发过程 使用特定的方式来进行配置（properties 或 yml 文件）
2. 创建独立的 spring 引用程序 main 方法运行
3. 嵌入的 Tomcat 无需部署 war 文件
4. 简化 maven 配置

2、什么是 JavaConfig?

Spring JavaConfig 是 Spring 社区的产品，它提供了配置 Spring IoC 容器的纯 Java 方法。因此它有助于避免使用 XML 配置。 使用 JavaConfig 的优点在于： 面向对象的配置。由于配置被定义为 JavaConfig 中的类，因此用户可以充分利用 Java 中的面向对象功能。一个配置类可以继承另一个，重写它的 @Bean 方法等。 减少或消除 XML 配置。基于依赖注入原则的外化配置的好处已被证明。但是，许多开发人员不希望在 XML 和 Java 之间来回切换。JavaConfig 为开发人员提供了一种纯 Java 方法来配置与 XML 配置概念相似的 Spring 容器。从技术角度来讲，只使用 JavaConfig 配置类来配置容器是可行的，但实际上很多人认为将 JavaConfig 与 XML 混合匹配是理想的。 类型安全和重构友好。 JavaConfig 提供了一种类型安全的方法来配置 Spring 容器。由于 Java 5.0 对泛型的支持，现在可以按类型而不是按名称检索 bean，不需要任何强制转换或基于字符串的查找。

3、Spring Boot 有哪些优点？ 答：

1. 快速创建独立运行的 spring 项目与主流框架集成
2. 使用嵌入式的 servlet 容器，应用无需打包成 war 包
3. starters 自动依赖与版本控制
4. 大量的自动配置，简化开发，也可修改默认值
5. 准生产环境的运行应用监控
6. 与云计算的天然集成

4、Spring Boot 提供了哪些核心功能?

1. 独立运行 Spring 项目
2. 内嵌 Servlet 容器 Spring Boot 可以选择内嵌 Tomcat、Jetty 或者 Undertow，这样我们无须以 war 包形式部署项目。
3. 提供 Starter 简化 Maven 配置 例如，当你使用了 spring-boot-starter-web ，会自动加入如下依赖：`spring-boot-starter-web` 的 pom 文件
4. 自动配置 Spring Bean Spring Boot 检测到特定类的存在，就会针对这个应用做一定的配置，进行自动配置 Bean ，这样会极大地减少我们要使用的配置。
5. 准生产的应用监控 Spring Boot 提供基于 HTTP、JMX、SSH 对运行时的项目进行监控。
6. 无代码生成和 XML 配置 Spring Boot 没有引入任何形式的代码生成，它是使用的 Spring 4.0 的条件 @Condition 注解以实现根据条件进行配置。同时使用了 Maven /Gradle 的依赖传递解析机制来实现 Spring 应用里面的自动配置。

5、如何重新加载 Spring Boot 上的更改，而无需重新启动服务器?

这可以使用 DEV 工具来实现。通过这种依赖关系，您可以节省任何更改，嵌入式 tomcat 将重新启动。 Spring Boot 有一个开发工具（DevTools）模块，它有助于提高开发人员的生产力。Java 开发人员面临的一个主要挑战是将文件更改自动部署到服务器并自动重启服务器。 开发人员可以重新加载 Spring Boot 上的更改，而无需重新启动服务器。这将消除每次手动部署更改的需要。Spring Boot 在发布它的第一个版本时没有这个功能。 这是开发人员最需要的功能。DevTools 模块完全满足开发人员的需求。该模块将在生产环境中被禁用。它还提供 H2 数据库控制台以更好地测试应用程序。

6、创建一个 Spring Boot Project 的最简单的方法是什么?

Spring Initializer 是创建 Spring Boot Projects 的一个很好的工具

7、运行 Spring Boot 有哪几种方式?

1. 打包成 Fat Jar ，直接使用 java -jar 运行。目前主流的做法，推荐。
2. 在 IDEA 或 Eclipse 中，直接运行应用的 Spring Boot 启动类的 #main(String[] args) 启动。适用于开发调试场景。
3. 如果是 Web 项目，可以打包成 War 包，使用外部 Tomcat 或 Jetty 等容器。

8、Spring Boot 中的监视器是什么?

Spring boot actuator 是 spring 启动框架中的重要功能之一。Spring boot 监视器可帮助您访问生产环境中正在运行的应用程序的当前状态。 有几个指标必须在生产环境中进行检查和监控。即使一些外部应用程序可能正在使用这些服务来向相关人员触发警报消息。监视器模块公开了一组可直接作为 HTTP URL 访问的 REST 端点来检查状态。

9、什么是 starter?

Starter 主要是用来简化 maven 依赖

10、Spring Boot 常用的 Starter 有哪些?

spring-boot-starter-web ： 提供 Spring MVC + 内嵌的 Tomcat 。 spring-boot-starter-data-jpa ： 提供 Spring JPA + Hibernate 。 spring-boot-starter-data-redis ： 提供 Redis 。 mybatis-spring-boot-starter ： 提供 MyBatis 。

11、什么是 YAML?

YAML 是一种人类可读的数据序列化语言。它通常用于配置文件。 与属性文件相比，如果我们想要在配置文件中添加复杂的属性，YAML 文件就更加结构化，而且更少混淆。可以看出 YAML 具有分层配置数据。

12、如何集成 Spring Boot 和 ActiveMQ?

对于集成 Spring Boot 和 ActiveMQ，我们使用 spring-boot-starter-activemq 依赖关系。 它只需要很少的配置，并且不需要样板代码。

13、springboot 常用的 starter 有哪些?

spring-boot-starter-web 嵌入 tomcat 和 web 开发需要 servlet 与 jsp 支持 spring-boot-starter-data-jpa 数据库支持 spring-boot-starter-data-redis redis 数据库支持 spring-boot-starter-data-solr solr 支持 mybatis-spring-boot-starter 第三方的 mybatis 集成 starter

14、springboot 自动配置的原理

在 spring 程序 main 方法中 添加@SpringBootApplication 或者 @EnableAutoConfiguration 会自动去 maven 中读取每个 starter 中的 spring.factories 文件 该文件里配置了所有需要被创建 spring 容器中的 bean

15、springboot 读取配置文件的方式

springboot 默认读取配置文件为 application.properties 或者是 application.yml

16、Spring Boot 需要独立的容器运行吗?

可以不需要，内置了 Tomcat/ Jetty 等容器。

17、运行 Spring Boot 有哪几种方式?

1）打包用命令或者者放到容器中运行

2）用 Maven/ Gradle 插件运行

3）直接执行 main 方法运行

18、Spring Boot 的核心配置文件有哪几个？它们的区别是什么？

Spring Boot 的核心配置文件是 application 和 bootstrap 配置文件。

application 配置文件这个容易了解，主要用于 Spring Boot 项目的自动化配置。 bootstrap 配置文件有以下几个应用场景。 使用 Spring Cloud Config 配置中心时，这时需要在 bootstrap 配置文件中增加连接到配置中心的配置属性来加载外部配置中心的配置信息； 少量固定的不能被覆盖的属性； 少量加密/解密的场景；

19、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： @SpringBootConfiguration：组合了 @Configuration 注解，实现配置文件的功能。 @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 })。 @ComponentScan：Spring 组件扫描

20、为什么我们需要 spring-boot-maven-plugin？

spring-boot-maven-plugin 提供了一些像 jar 一样打包或者运行应用程序的命令。 spring-boot:run 运行你的 SpringBooty 应用程序。 spring-boot：repackage 重新打包你的 jar 包或者是 war 包使其可执行 spring-boot：start 和 spring-boot：stop 管理 Spring Boot 应用程序的生命周期（也可以说是为了集成测试）。 spring-boot:build-info 生成执行器可以使用的构造信息。

21、如何使用 Spring Boot 实现分页和排序?

使用 Spring Boot 实现分页非常简单。使用 Spring Data-JPA 可以实现将可分页的 org.springframework.data.domain.Pageable 传递给存储库方法。

22、什么是 Swagger? 你用 Spring Boot 实现了它吗?

Swagger 广泛用于可视化 API，使用 Swagger UI 为前端开发人员提供在线沙箱。Swagger 是用于生成 RESTful Web 服务的可视化表示的工具，规范和完整框架实现。它使文档能够以与服务器相同的速度更新。当通过 Swagger 正确定义时，消费者可以使用最少量的实现逻辑来理解远程服务并与其进行交互。因此，Swagger 消除了调用服务时的猜测。

23、什么是 Spring Profiles?

Spring Profiles 允许用户根据配置文件（dev，test，prod 等）来注册 bean。因此，当应用程序在开发中运行时，只有某些 bean 可以加载，而在 PRODUCTION 中，某些其他 bean 可以加载。假设我们的要求是 Swagger 文档仅适用于 QA 环境，并且禁用所有其他文档。这可以使用配置文件来完成。Spring Boot 使得使用配置文件非常简单。

24、什么是 Spring Batch?

Spring Boot Batch 提供可重用的函数，这些函数在处理大量记录时非常重要，包括日志/跟踪，事务管理，作业处理统计信息，作业重新启动，跳过和资源管理。它还提供了更先进的技术服务和功能，通过优化和分区技术，可以实现极

高批量和高性能批处理作业。简单以及复杂的大批量批处理作业可以高度可扩展的方式利用框架处理重要大量的信息。

25、什么是 FreeMarker 模板?

FreeMarker 是一个基于 Java 的模板引擎，最初专注于使用 MVC 软件架构进行动态网页生成。使用 Freemarker 的主要优点是表示层和业务层的完全分离。程序员可以处理应用程序代码，而设计人员可以处理 html 页面设计。最后使用 freemarker 可以将这些结合起来，给出最终的输出页面。

26、什么是 JavaConfig?

Spring JavaConfig 是 Spring 社区的产品，它提供了配置 Spring IoC 容器的纯 Java 方法。因此它有助于避免使用 XML 配置。使用 JavaConfig 的优点在于： 面向对象的配置。由于配置被定义为 JavaConfig 中的类，因此用户可以充分利用 Java 中的面向对象功能。一个配置类可以继承另一个，重写它的@Bean 方法等。 减少或消除 XML 配置。基于依赖注入原则的外化配置的好处已被证明。但是，许多开发人员不希望在 XML 和 Java 之间来回切换。 JavaConfig 为开发人员提供了一种纯 Java 方法来配置与 XML 配置概念相似的 Spring 容器。 从技术角度来讲，只使用 JavaConfig 配置类来配置容器是可行的，但实际上很多人认为将 JavaConfig 与 XML 混合匹配是理想的。 类型安全和重构友好。JavaConfig 提供了一种类型安全的方法来配置 Spring 容器。由于 Java 5.0 对泛型的支持，现在可以按类型而不是按名称检索 bean，不需要任何强制转换或基于字符串的查找

27、启动类注解：

@SpringBootConfiguration:Spring Boot 的配置类；标注在某个类上，表示这是一个 Spring Boot 的配置类; @Configuration:配置类上来标注这个注解;配置类 -----配置文件;配置类也是容器中的一个组件;@Component

@EnableAutoConfiguration:开启自动配置功能；以前我们需要配置的东西，Spring Boot 帮我们自动配置;@EnableAutoConfiguration 告诉 SpringBoot 开启自动配置功能;这样自动配置才能生效; Spring Boot 在启动的时候从类路径下的 META-INF/spring.factories 中获取 EnableAutoConfiguration 指定的值，将这些值作为自动配置类导入到容器中，自动配置类就失效，帮我们进行自动配置工作

28、配置文件的加载顺序

由 jar 包外向 jar 包内进行寻找；优先加载带 profile jar 包外部的 application-{profile}.properties 或 application.yml(带 spring.profile)配置文件 jar 包内部的 application-{profile}.properties 或 application.yml(带 spring.profile)配置文件 再来加载不带 profile jar 包外部的 application.properties 或 application.yml(不带 spring.profile)配置文件 jar 包内部的 application.properties 或 application.yml(不带 spring.profile)配置文件

29、自动配置原理

1)、SpringBoot 启动的时候加载主配置类，开启了自动配置功能 @EnableAutoConfiguration

2)、@EnableAutoConfiguration 作用：将类路径下 META-INF/spring.factories 里面配置的所有 EnableAutoConfiguration 的值加入到了容器中；每一个这样的 xxxAutoConfiguration 类都是容器中的一个组件，都加入到容器中;用他们来做自动配置;

3)、每一个自动配置类进行自动配置功能；根据当前不同的条件判断，决定这个配置类是否生效；

4)、一但这个配置类生效;这个配置类就会给容器中添加各种组件;这些组件的属性是从对应的 properties 类中获取 的，这些类里面的每一个属性又是和配置文件绑定的;

5)、所有在配置文件中能配置的属性都是在 xxxxProperties 类中封装者‘;配置文件能配置什么就可以参照某个功 能对应的这个属性类

30、怎么用好自动配置，精髓:

1)、SpringBoot 启动会加载大量的自动配置类

2)、我们看我们需要的功能有没有 SpringBoot 默认写好的自动配置类;

3)、我们再来看这个自动配置类中到底配置了哪些组件;(只要我们要用的组件有，我们就不需要再来配置了)

4)、给容器中自动配置类添加组件的时候，会从 properties 类中获取某些属性。我们就可以在配置文件中指定这 些属性的值;

31、日志框架：

SpringBoot 选用 SLF4j 和 logback; 如何让系统中所有的日志都统一到 slf4j;

1、将系统中其他日志框架先排除出去;

2、用中间包来替换原有的日志框架;

3、我们导入 slf4j 其他的实现 SpringBoot 能自动适配所有的日志，而且底层使用 slf4j+logback 的方式记录日志

32、Spring Boot、Spring MVC 和 Spring 有什么区别?

Spring 是一个“引擎”， Spring MVC 是基于 Spring 的一个 MVC 框架， Spring Boot 是基于 Spring 的一套快速开发整合包

33、我们如何监视所有 Spring Boot 微服务?

Spring Boot 提供监视器端点以监控各个微服务的度量。这些端点对于获取有关应用程序的信息（如它们是否已启动）以及它们的组件（如数据库等）是否正常运行很有帮助。但是，使用监视器的一个主要缺点或困难是，我们必须单独打开应用程序的知识点以了解其状态或健康状况。想象一下涉及 50 个应用程序的微服务，管理员将不得不击中所有 50 个应用程序的执行终端。