# 1、什么是 Spring Boot？

Spring Boot 是 Spring 开源组织下的子项目，是 Spring 组件一站式解决方案，主要是简化了使用 Spring 的难度，简省了繁重的配置，提供了各种启动器，开发者能快速上手。

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

5.自动配置spring添加对应功能starter自动化配置

# 2、为什么要用 Spring Boot？

Spring Boot 优点非常多，如：

独立运行

简化配置

自动配置

无代码生成和XML配置

应用监控

上手容易

…

Spring Boot 集这么多优点于一身，还有理由不使用它呢？

# 3、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 配置文件这个容易理解，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 配置文件中添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；
* 一些固定的不能被覆盖的属性；
* 一些加密/解密的场景；

# 4、Spring Boot 的配置文件有哪几种格式？它们有什么区别？

.properties 和 .yml，它们的区别主要是书写格式不同。

1).properties

app.user.name = javastack

2).yml

app:

  user:

    name: javastack

另外，.yml 格式不支持 @PropertySource 注解导入配置。

# 5、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。

# 6、开启 SpringBoot特性有哪几种方式？

1）继承spring-boot-starter-parent项目

2）导入spring-boot-dependencies项目依赖

具体请参考这篇文章《[Spring Boot开启的2种方式](https://blog.csdn.net/zl1zl2zl3/article/details/83715712)》。

# 7、SpringBoot需要独立的容器运行吗？

可以不需要，内置了 Tomcat/ Jetty 等容器。

# 8、运行SpringBoot有哪几种方式？

1）打包用命令或者放到容器中运行

2）用 Maven/ Gradle 插件运行

3）直接执行 main 方法运行

# 9、SpringBoot自动配置原理是什么？

注解 @EnableAutoConfiguration, @Configuration, @ConditionalOnClass 就是自动配置的核心，首先它得是一个配置文件，其次根据类路径下是否有这个类去自动配置。

具体看这篇文章《[Spring Boot自动配置原理、实战](https://blog.csdn.net/zl1zl2zl3/article/details/83715154)》。

# 10、SpringBoot的目录结构是怎样的？

1. cn
2. +- javastack
3. +- MyApplication.java
4. |
5. +- customer
6. |   +- Customer.java
7. |   +- CustomerController.java
8. |   +- CustomerService.java
9. |   +- CustomerRepository.java
10. |
11. +- order
12. +- Order.java
13. +- OrderController.java
14. +- OrderService.java
15. +- OrderRepository.java

这个目录结构是主流及推荐的做法，而在主入口类上加上 @SpringBootApplication 注解来开启 Spring Boot 的各项能力，如自动配置、组件扫描等。具体看这篇文章《[Spring Boot 主类及目录结构介绍](https://blog.csdn.net/zl1zl2zl3/article/details/83715763)》。

# 11、你如何理解SpringBoot中的Starters？

Starters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 JPA 访问数据库，只要加入 spring-boot-starter-data-jpa 启动器依赖就能使用了。

Starters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。具体请看这篇文章《[Spring Boot Starters启动器](https://blog.csdn.net/zl1zl2zl3/article/details/83715784)》。

# 12、如何在Spring Boot启动的时候运行一些特定的代码？

可以实现接口 ApplicationRunner 或者 CommandLineRunner，这两个接口实现方式一样，它们都只提供了一个 run 方法，具体请看这篇文章《[Spring Boot Runner启动器](https://blog.csdn.net/zl1zl2zl3/article/details/83715818)》。

# 13、Spring Boot有哪几种读取配置的方式？

Spring Boot 可以通过 @PropertySource,@Value,@Environment, @ConfigurationProperties 来绑定变量，具体请看这篇文章《[Spring Boot读取配置的几种方式](https://blog.csdn.net/zl1zl2zl3/article/details/83715873)》。

# 14、SpringBoot支持哪些日志框架？推荐和默认的日志框架是哪个？

Spring Boot 支持 Java Util Logging, Log4j2, Lockback 作为日志框架，如果你使用 Starters 启动器，Spring Boot 将使用 Logback 作为默认日志框架，具体请看这篇文章《[Spring Boot日志集成](https://blog.csdn.net/zl1zl2zl3/article/details/83715942)》。

# 15、SpringBoot实现热部署有哪几种方式？

主要有两种方式：

* Spring Loaded
* Spring-boot-devtools

Spring-boot-devtools 使用方式可以参考这篇文章《[Spring Boot实现热部署](https://blog.csdn.net/zl1zl2zl3/article/details/83716502)》。

# 16、你如何理解 Spring Boot 配置加载顺序？

在 Spring Boot 里面，可以使用以下几种方式来加载配置。

1）properties文件；

2）YAML文件；

3）系统环境变量；

4）命令行参数；

等等……

具体请看这篇文章《[Spring Boot 配置加载顺序详解](https://blog.csdn.net/zl1zl2zl3/article/details/83716564)》。

# 17、Spring Boot 如何定义多套不同环境配置？

提供多套配置文件，如：

1. applcation.properties
3. application-dev.properties
5. application-test.properties
7. application-prod.properties

运行时指定具体的配置文件，具体请看这篇文章《[Spring Boot Profile 不同环境配置](https://blog.csdn.net/zl1zl2zl3/article/details/83716663)》。

# 18、Spring Boot 可以兼容老 Spring 项目吗，如何做？

可以兼容，使用 @ImportResource 注解导入老 Spring 项目配置文件。

# 19、保护 Spring Boot 应用有哪些方法？

* 在生产中使用HTTPS
* 使用Snyk检查你的依赖关系
* 升级到最新版本
* 启用CSRF保护
* 使用内容安全策略防止XSS攻击
* …

更多请看这篇文章《[10 种保护 Spring Boot 应用的绝佳方法](https://blog.csdn.net/zl1zl2zl3/article/details/83716770)》。

# 20、Spring Boot 2.X 有什么新特性？与 1.X 有什么区别？

* 配置变更
* JDK 版本升级
* 第三方类库升级
* 响应式 Spring 编程支持
* HTTP/2 支持
* 配置属性绑定
* 更多改进与加强…

# 21.springboot与spring的区别.

引用自官方说法: java在集成spring等框架需要作出大量的配置,开发效率低,繁琐.所以官方提出 spring boot的核心思想:习惯优于配置.可以快速创建开发基于spring框架的项目.或者支持可以不用或很少的spring配置即可.

# 22.springboot的核心功能与使用优点.

**核心功能:**1.1: springboot项目为独立运行的spring项目,java -jar xx.jar即可运行.  
1.2: 内嵌servlet容器(可以选择内嵌: tomcat ,jetty等服务器.).  
1.3: 提供了starter的pom 配置 简化了maven的配置.  
1.4: 自动配置spring容器中的bean.当不满足实际开发场景,可自定义bean的自动化配置.  
1.5: 准生产的应用监控(基于: ssh , http , telnet 对服务器运行的项目进行监控.).  
1.6: springboot无需做出xml配置,也不是通过代码生成来实现(通过条件注解)

**使用优点:**1.快速搭建项目,  
2,与主流框架集成无需配置集成.  
3.内嵌服务容器.  
4.具有应用监控.  
5.开发部署方便,后期与云计算平台集成方便(docket).

# 23.springboot中的application.properties配置文件是什么,有哪些配置.

application.properties为boot项目中的一个系统自带的全局属性配置文件. 提供默认属性重写的作用. 可包含重写系tomcat,spring,springmvc,mybatis等诸多默认配置属性: 列举部分如下:  
#全局配置文件: 重写视图解析器的资源地址.  
#页面默认前缀目录  
spring.mvc.view.prefix=/WEB-INF/jsp/  
#?响应页面默认后缀  
spring.mvc.view.suffix=.jsp  
#静态资源目录配置,  
spring.mvc.static-path-pattern=/static/\*\*

#tomcat服务器的配置:  
server.port=8081  
server.servlet.context-path=/sb2

#默认支持的日志记录:  
#logging.config=classpath:logback.xml 加载单独的日志配置文件.  
logging.file=d:/test/log.log  
logging.level.org.springframework.web=DEBUG

#提供jdbc的基本配置:  
spring.datasource.driver-class-name=com.mysql.jdbc.Driver  
spring.datasource.url=jdbc:mysql://localhost:3306/c01?useUnicode=true&characterEncoding=utf-8  
spring.datasource.username=root  
spring.datasource.password=root  
spring.datasource.type=org.apache.commons.dbcp.BasicDataSource  
#提供mybatis的属性配置: 扫描.  
mybatis.mapper-locations=classpath:mapper/\*\_mapper.xml

# 24.springboot中常用的starter的组件有哪些.

spring-boot-starter-parent //boot项目继承的父项目模块.  
spring-boot-starter-web //boot项目集成web开发模块.  
spring-boot-starter-tomcat //boot项目集成tomcat内嵌服务器.  
spring-boot-starter-test //boot项目集成测试模块.  
mybatis-spring-boot-starter //boot项目集成mybatis框架.  
spring-boot-starter-jdbc //boot项目底层集成jdbc实现数据库操作支持.  
其他诸多组件,可到maven中搜索,或第三方starter组件到github上查询 .....

# 25.springboot中的核心启动主函数(main函数)的作用.用到哪些注解.注解的作用.

@SpringBootApplication  
public class SpringBoot1Application {  
public static void main(String[] args) {  
SpringApplication.run(SpringBoot1Application.class, args);  
}  
}  
该主函数: 主要启动springboot框架.用于加载容器和诸多默认组件.  
用到核心注解: @SpringBootApplication . 作用:用于标识声明一个springboot框架容器.

# 26.springboot中的常用配置入口有哪些?

bootstrap.properties/bootstrap.yml //用于配置无需重写的系统常量,例如springcloud项目用到的config配置中心的连接属性等.加载优先级高于application.properties.

application.properties/application.yml //用于配置重写springboot项目的默认框架属性,例如:重写tomcat,springmvc,日志框架等默认属性.主要提供给spring框架加载使用.

注: properties后缀名与yml后缀名配置文件二选一即可. 两种不同格式的配置文件而已.

# 27.springboot项目需要兼容老项目（spring框架）,该如何实现.

集成老项目spring框架的容器配置文件即可:  
spring-boot一般提倡零配置.但是如果需要配置,也可增加:  
@ImportResource({"classpath:spring1.xml" , "classpath:spring2.xml"})  
注意:resources/spring1.xml位置.

# 28.需要加载外部配置文件中的自定义属性,该如何实现.

需求一批量加载多个属性.  
步骤一: 首先需要自定义外部配置文件和其中的自定义属性:  
user.properties . 存放在resources目录下:  
内部:  
#自定义配置其他属性:  
user.username=zhangsan  
user.age=20  
步骤二: 加载属性到程序中:  
springboot 1.5版本以及之前采用:[br/>@ConfigurationProperties(prefix="user",locations={"classpath:user.propeties"})  
<="" a="">public class User {  
private String username;  
private Integer age;  
get/set封装省略....  
}  
springboot 1.5版本以后采用如下:   
@PropertySource(value ="classpath:user.properties")  
@ConfigurationProperties(prefix = "user")](mailto:br/%3e@ConfigurationProperties(prefix=)[br/>@Component  
<="" a="">public class User {  
private String username;  
private Integer age;  
get/set封装省略....  
}  
步骤三:  
以上配置需要在main启动函数类文件上激活配置: 新版本中默认开启.  
@EnableConfigurationProperties](mailto:br/%3e@Component%3cbr/)

[需求二:如果加载单个属性:  
步骤一：省略.如上.  
步骤二:](mailto:br/%3e@Component%3cbr/) [br/>@Value("${name}")  
<="" a="">private String name;.](mailto:br/%3e@Value()

[备注:以上外部文件属性加载.切记注意中文乱码问题.](mailto:br/%3e@Value()

# [29.springboot支持的默认日志框架有哪些.可以进行哪些设置.](mailto:br/%3e@Value()

[spring-boot: 默认采用Logback作为日志框架.  
配置即可:  
logging.file=d:/test/log.log  
logging.level.org.springframework.web=DEBUG  
#logging.config=classpath:logback.xml 加载单独的日志配置文件.  
其他配置项......](mailto:br/%3e@Value()

# [30.springboot项目的开发环境,生产环境配置该如何实现切换.](mailto:br/%3e@Value()

[profile配置:](mailto:br/%3e@Value()

[spring-boot默认为了支持不同的配置环境.](mailto:br/%3e@Value()

[配置步骤: 1.提供环境:](mailto:br/%3e@Value()

[按照命名模板:application-{profile}.properties(例如: application-pro1.properties/application-pro2.properties)](mailto:br/%3e@Value()

[2.选择激活的环境:](mailto:br/%3e@Value()

[application.properties中设置:spring.profiles.active=pro1](mailto:br/%3e@Value()