JAR包和WAR包是两种常见的Java应用程序打包格式，它们主要用于不同类型的应用部署和分发。

JAR包（Java Archive）：JAR包是一种用于打包Java类、资源文件和库文件的归档文件格式。它通常用于打包和分发独立的Java应用程序、库或模块。JAR包可以包含可执行的Java程序（例如main方法所在的类）、类文件、资源文件、配置文件和第三方库等。JAR包可以直接通过java -jar命令运行。

WAR包（Web Application Archive）：WAR包是一种用于打包和部署Java Web应用程序的归档文件格式。它是一种特殊的JAR包，用于打包Web应用程序的相关资源和文件。WAR包通常包含Web应用程序的HTML、CSS、JavaScript等静态资源文件，以及Servlet、JSP、配置文件、标签库等动态资源文件。WAR包可以被部署到支持Java Web容器（如Tomcat、Jetty等）的服务器上运行。

总结来说，JAR包主要用于打包和分发独立的Java应用程序和库，而WAR包则专门用于打包和部署Java Web应用程序。它们的区别在于打包的内容和针对的应用类型。

1、以jar方式来部署

如果要以jar方式来部署spring boot应用，只需要加入下面的插件即可

<build>

   <plugins>

       <plugin>

           <groupId>org.springframework.boot</groupId>

           <artifactId>spring-boot-maven-plugin</artifactId>

           <configuration>

               <executable>true</executable>

           </configuration>

       </plugin>

   </plugins>

</build>

接着只要执行：mvn package，即可完成打包，这是一个fatjar包，包含了所有依赖，可以执行的一个jar包

接着可以将jar包通过scp命令上传到服务器上

执行jar包启动系统：java -jar target\springboot-demo-1.0.0.jar

2、以war包方式来部署

一般来说，实际去部署的时候，都不会是以jar包方式部署的，因为那是用的spring boot内嵌的tomcat服务器来部署的，一般还是以war包方式，部署到线上已有的tomcat容器中的

此时要做的第一件事情，就是将spring boot应用的pom.xml中的packaging从jar修改为war

咱们阶段一，先用spring boot内嵌容器去部署的方式，上线，因为一开始假设的是这个系统几乎就没什么用人，所以哪怕是默认的配置，其实支撑少量的请求，都是没什么问题的

但是到了阶段二，就是已经开始有一定的访问量了，那么那个时候开始，我们就会正经使用tomcat去部署，而且会深入的学习tomcat，内核

<groupId>com.zhss.springboot</groupId>

<artifactId>springboot-demo</artifactId>

<packaging>war</packaging>

接着需要将spring boot内嵌的tomcat修改为provided范围

<dependency>

   <groupId>org.springframework.boot</groupId>

   <artifactId>spring-boot-starter-tomcat</artifactId>

   <scope>provided</scope>

</dependency>

接着要再次修改Application类

// 这里必须继承SpringBootServletInitializer基类

@SpringBootApplication

public class Application extends SpringBootServletInitializer {

   // 这里必须实现一个configure方法

   @Override

   protected SpringAppliciationBuilder configure(SpringApplicationBuilder application) {

       return application.sources(Application.class);

  }

   public static void main(String[] args) {

       SpringApplication.run(Application.class, args);

  }

}

然后再次使用mvn package打包，就是一个war包

然后将war包放到tomcat的webapps目录中，启动tomcat，即可完成系统的部署

3、多环境部署

可以在src/main/resources目录下，放多个application-{profile}.properties文件

然后在启动的时候，使用java -jar -Dspring.profiles.active=prod target\springboot-demo-1.0.0.jar来指定要激活哪个环境的profile

在实际的项目中，你永远无法保证说，你多环境下，就是一个properties就可以了，我告诉大家一点，很多大公司里，自己研发的设施，是要求你不同的环境必须放不同的一个文件，让他们去读取的，跟spring boot是不兼容的。META-INF/conf.properties

我们用的是maven profile，我们在不同profile的路径下，放一个application.properites，以及也许需要放置的其他的一些配置文件，为了以防万一

也可以在tomcat的catalina.sh中，编辑系统属性来激活对应的profile，比如JAVA\_OPTS="-Dspring.profiles.active=prod"

还可以使用@Profile来标准@Configuration，来决定不同的环境下，可以让不同的类生效

但是实际项目中，还是建议使用maven的profile支持多环境吧，更加灵活。毕竟，spring boot的多环境支持，仅仅能支持多个application.properties，那如果还有别的配置文件呢？？

853769620