Day 1

1. 单体应用架构和微服务架构的区别：

（1）单体应用架构是把多个功能放在一个项目里面；

（2）微服务架构：一个项目可以由多个小型服务构成，每个小型服务实现一个功能，通过http协议进行连接，此时每个小型服务就是一个微服务。spring boot可以快速开发微服务；

2. 创建一个应用需要以下三步：

（1）使用spring boot创建并实现微服务；

（2）使用spring cloud来解决微服务之间的合作；

（3）使用spring cloud data flow来实现多个微服务的并行计算；

3. spring boot的优点：

（1）可以简化j2ee的开发；

（2）是整个spring技术栈的整合（如Spring Data;Spring Session;Spring Batch）；

（3）是整个j2ee技术的大整合（如Redis,Mybatis）；

4. 开发微服务的工具（建议使用STS或IDEA）：

（1）eclipse（需要安装STS插件）；

（2）STS（相当于安装了STS插件的eclipse）：百度STS，找到Dawnload STS并点击进去；

（3）IDEA

5. 使用spring boot创建微服务：

（1）安装配置JDK；

（2）安装配置Maven；

（3）在IDE（如STS）里面配置JDE及Maven（跟eclipse的配置方法相同）；

（4）进入http://spring.io/ --> 点击“PROJECTS” --> 点击SPRING BOOT --> 点击Quick start下面的“Spring Initializr” --> 进入后可以创建一个maven项目，具体如下：

a. Generate a Maven Project with Java and Spring Boot 2.0.3;

b. Group: org.gls

c. Artifact: HelloWorld

d. Search for dependencies：搜索依赖（可以点击“Switch to the full version”参考）

e. 点击“Generate Project”生成项目；

（5）在STS里面引入一个存在的maven项目，指定到刚刚生成的这个项目；

（6）运行“/HelloWorld/src/main/java/org/gls/HelloWorld/HelloWorldApplication.java”，在Console窗口看到Spring 标识，以及Spring Boot 的版本号，则说明Spring 已经配置好了。

6. STS里面的Spring Boot项目的目录结构说明如下：

（1）src/main/java：代码；

（2）jrc/main/resources：资源或配置文件；

a. static：静态资源（js , css , 图片, 音频, 视频）；

b. templates：模板文件（freemarker, thymeleaf） -- Spring Boot 默认情况下不支付JSP

c. application.properties：配置文件；

7. 传统WEB程序与用Spring boot实现WEB程序的区别：

（1）传统的WEB程序：需要在webContext里面配置web.xml文件，需要打war包，以及把war包放到tomcat里面去运行；

（2）用Spring Boot实现WEB程序时上面的三点都不需要（web.xml,war,tomcat）；因为SpringBoot内置了tomcat，且不需要打成war包执行；

8. 使用Spring Boot实现一个WEB程序，具体如下：

（1）在已经配置好的Spring Boot项目里面添加文件“HelloWorldController.java”，具体见“/HelloWorld/src/main/java/org/gls/HelloWorld/Controller/HelloWorldController.java”；

（2）运行Spring boot程序（即运行HelloWorldApplication）；

（3）在浏览器地址栏输入http://localhost:8080/helloWorld即可；

9. 在application.properties配置文件里面可以进行如下配置：

（1）server.port=8888 对端口号进行配置；

（2）

（3）

（4）

（5）

10. spring boot的pom文件解析：

（1）文件“/HelloWorld/pom.xml”的父类的父类

“org.springframework.boot:spring-boot-dependencies:2.1.1.RELEASE.pom”里面把所有依赖的版本号都定义好了，所以项目里面添加依赖时不需要再指定版本号（这样做可以避免版本冲突）；

（2）通过以下依赖引入WEB场景：

<dependency>

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

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

</dependency>

（3）进入http://spring.io/ --> 点击“PROJECTS” --> 点击SPRING BOOT --> 点击Learn --> 点击2.1.1对应的“Reference Doc.” --> 点击“13.5. Starters”，就可以看到Spring Boot所提供的所有场景（选择某个场景，就会把该场景下的所有依赖一起引入）；

（4）spring boot 将各个应用或者第三方框架设置成了一个个场景starter，以后要用哪个，只需要引入那个场景即可；选完之后spring boot就会将该场景所需要的所有依赖自动注入。例如选把“web”，spring boot就会将web相关的依赖（如tomcat,json）全部引入本项目。

11. @SpringBootApplication注解讲解如下：

（1）这个注解是文件“/HelloWorld/src/main/java/org/gls/HelloWorld/HelloWorldApplication.java”里面的主配置类；

（2）该注解包含很多注解，重点看下面两个：

a. @SpringBootConfiguration：包含@Configuration，表示“配置类”，作用如下：

a) 表示该类是一个配置类

b) 加了该注解的类，会自动纳入Spring容器

b. @EnableAutoConfiguration：使spring boot可以自动配置，可以找到@SpringBootApplication所在类的包，作用是将该包及所有的子包，全部纳入spring容器内；

（3）

第03课 05:19