## 1.什么是Springboot？

SpringBoot是一个快速开发的框架，能过快速帮我们整合常用的第三方框架;

## 2.Springboot如何整合第三方框架的？

基于Maven的继承方式,完全采用注解化(使用注解的方式开启SpringMVC),简化XML配置，内置HTTP服务器（Tomcat，Jetty），最终以java应用程序进行执行；无需发布打包到服务器,直接通过Java命令即可执行(java -jar)

## 3.为什么使用Springboot？

**传统项目：**

a.整合SSH或SSM，考虑 配置文件，jar包冲突问题,整合起来繁琐;

打包方式：打成一个war包放入到tomacat的webapps目录下进行执行;

**Springboot项目：**

a.Springboot 是一个快速开发框架，能够帮助我们快速整合第三方常用框架（Maven依赖关系（Maven的继承）），完全使用注解化，简化xml文件配置(在spring3.0之后采用注解方式启动Springmvc),内置嵌入了Http服务器（Tomcat,Jetty)，默认嵌入了Tomacat服务器;最终以java应用程序进行执行;使用Java创建Tomacat;Springboot项目没有web.xml;

## 4.Springboot与SpringMVC的关系？

a.Springboot web组件集成了SpringMvc框架;但是Springboot启动SpringMVc是没有传统的配置文件;

b.SpringBoot如何启动SpringMvc？

在SpringMVC3.0支持注解方式启动;

## 5. Springboot与SpringCould的关系？

a.Springboot是一个快速开发框架，可以快速帮助我们整理常用第三方框架；

完全采用注解化，简化xml文件配置，最终以java应用程序进行执行;

b.SpringCould是一套目前完整微服务解决框架;

注册中心，客户端调用工具，服务治理（负载均衡，断路器，分布式配置中心，网关，服务链路等）;

**c.关系：**

**Springboot是 微服务框架？**

SpringBoot+SpringCould 实现微服务开发;

SpringBoot实现快速开发;

微服务通讯技术http+json(restful)轻量级Springboot Web组件默认集成了SpringMVC，SpringCould依赖于Springboot实现微服务, 使用SpringMVC编写微服务接口;

## 6.springboot2.X自动装配原理（HttpEncodingAutoConfiguration为例）

@Configuration(proxyBeanMethods = **false**)

@EnableConfigurationProperties(HttpProperties.**class**)

@ConditionalOnWebApplication(type = ConditionalOnWebApplication.Type.***SERVLET***)

@ConditionalOnClass(CharacterEncodingFilter.**class**)

@ConditionalOnProperty(prefix = "spring.http.encoding", value = "enabled", matchIfMissing = **true**)

**public** **class** HttpEncodingAutoConfiguration {

}

@ConfigurationProperties(prefix = "spring.http")

**public** **class** HttpProperties {

}

1.@Configuration：标识此类为配置类,将此类纳入SpringIOC容器;

2.@EnableConfigurationProperties(HttpProperties.**class**):

通过HttpProperties将默认编码设置为UTF-8（自动装配为UTF-8）

3.修改默认编码

通过HttpProperties中的prefix+属性名（charset）进行修改[在配置文件中yml/properties, 该注解提供了默认的utf-8，并提供了prefix+属性名的方式来修改编码;在配置文件中配置; spring.http.encoding.charset=ISO-8859-1

4. @ConditionalOnWebApplication(type = ConditionalOnWebApplication.Type.***SERVLET***)

仅在当前应用是 Servlet Web 应用时才生效@ConditionalOnClass(CharacterEncodingFilter.**class**)

仅在类 CharacterEncodingFilter 存在于 classpath 上时才生效@ConditionalOnProperty(prefix = "spring.http.encoding", value = "enabled", matchIfMissing = **true**)

仅在配置项 spring.http.encoding.enabled 不存在或者设置为 true 时才生效

Conditional条件;on当...的时候;WebApplication web应用;

value:等价于name;

prefix:前缀;

matchIfMissing:如果没有匹配则满足;

Conditional其他注解:

@ConditionalOnMissingBean:容器中不存在指定Bean时生效;

@ConditionalOnBean:容器中存在指定Bean时生效;

@ConditionalOnMissingClass:容器中不存在指定类时生效;

@ConditionalOnClass:容器中存在指定类是生效;

@ConditionalOnNotWebApplication:当前不是web环境时生效;

@ConditionalOnWebApplication:当前是web环境时生效;

@ConditionalOnProperty:系统中指定的属性是否有指定的值;

@ConditionalOnJava:系统中java版本是否符合要求;

@ConditionalOnExpression:满足指定SpEl表达式;

@ConditionalOnSingleCandidate:容器中只有一个指定的Bean，或者这个Bean是首选Bean;

@ConditionalOnResource:类路径下存在指定的资源文件生效;

@ConditionalOnJndi:JNDI存在指定项;

总结：

1.每个XxAutoConfiguration都有很多@ConditionalOnXx,当这些条件满足是,则此配置自动生效; 但是可以手工修改自动装配XxxProperties中的prefix+属性名=value;

2.全局配置文件中的key,来源于某个XxxProperties中的prefix+属性名=value;

## 7.springboot2.2.x集成eureka

基本环境:springboot+maven+jdk1.8

步骤：1.创建eureka-server注册中心模块;

2.创建eureka-client服务提供者将服务注册到eureka-server;

### 添加依赖

<parent>

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

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

<version>2.2.13.RELEASE</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<properties>

<java.version>1.8</java.version>

<spring-cloud.version>Hoxton.M3</spring-cloud.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>

</dependency>

</dependencies>

<dependencyManagement>

<dependencies>

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-dependencies</artifactId>

<version>${spring-cloud.version}</version>

<type>pom</type>

<scope>import</scope>

</dependency>

</dependencies>

</dependencyManagement>

### 创建注册中心

server:

port: '@eureka.server.port@'

eureka:

instance:

hostname: '@eureka.host@' #注册中心IP地址

client:

service-url:

defaultZone: http://${eureka.instance.hostname}:${server.port}/eureka #注册地址

register-with-eureka: false #因为自己是注册中心，是否将自己注册给自己的注册中心（集群的时候需要设为true）

fetch-registry: false #因为自己是注册中心，不需要检索服务信息

spring:

profiles:

active: '@profiles.active@'

application:

name: tms-eureka-server

### 创建服务提供者

#端口

server:

port: '@server.account.port@'

spring:

profiles:

active: '@profiles.active@'

application:

name: tms-account-service

#eureka

eureka:

client:

service-url:

defaultZone: http://@eureka.host@:@eureka.server.port@/eureka

instance:

prefer-ip-address: '@eureka.instance.prefer-ip-address@'

## 8.springboot集成数据库Mysql

步骤：1.添加依赖

2.配置数据库信息

### 添加相关依赖

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.18</version>

</dependency>

### 配置数据库信息

spring:

datasource: #mysql的配置

url: '@spring.datasource.url@'

driver-class-name: '@spring.datasource.driverClassName@'

username: '@spring.datasource.username@'

password: '@spring.datasource.password@'

## 9.springboot集成数据库Oracle

步骤:

1.添加依赖

2.配置Oracle

### 添加依赖

<dependency>

<groupId>com.oracle</groupId>

<artifactId>ojdbc14</artifactId>

<version>${ojdbc14.version}</version>

</dependency>

Maven3没有提供Oracle JDBC driver，为了在Maven项目中应用Oracle JDBC driver,必须手动添加到本地仓库。

1. 下载Oracle的驱动包

mvn install:install-file -DgroupId=com.oracle -DartifactId=ojdbc14 -Dversion=10.2.0.4.0 -Dpackaging=jar -Dfile=D:\Oracle\_10g\_10.2.0.4\_JDBC\_ojdbc14.jar

1. 相关文档

<https://www.pianshen.com/article/8066985458/>

### 配置数据库信息

spring:

datasource: #Oracle的配置

url: '@spring.datasource.url@'

driver-class-name: '@spring.datasource.driverClassName@'

username: '@spring.datasource.username@'

password: '@spring.datasource.password@'

### 出现的问题：

1. java.lang.ClassNotFoundException: oracle.jdbc.driver.OracleDriver

可能原因：

1. **没有导入驱动jar包;**

## 10.springboot整合Mybatis plus

步骤：

1. 添加依赖;
2. 配置Mybatis plus

### 添加依赖

<dependency>

<groupId>com.baomidou</groupId>

<artifactId>mybatis-plus-boot-starter</artifactId>

<version>${mybatis-plus.version}</version>

</dependency>

### 配置Mybatis plus

Yml:

## 11.Springboot整合MyBatis-Plus 的代码生成器AutoGenerator

1. 添加依赖
2. 编写代码