@RestController

用WEB的方式访问类

@RequestMapping(value = "/hello",method = RequestMethod.*GET*)

访问的目录，用GET的方式访问

mvn spring-boot:run

进入项目目录，用以上命令也可以启动网站

Mvn install

编译一下程序

Cd target

进入target目录

Java –jar girl-0.0.1-snapshot.jar

找到jar文件，运行

Application.yml 配置文件，如：

**server:  
 port:** 8081  
 **context-path:** "/context"  
  
**cupSize:** B  
**age:** 18  
**Content:** "cupSize:${**cupSize**} age:${**age**}}"

用以下方法使用（但一般不会这样用）

@RestController  
public class HelloController {  
  
 @Value("${cupSize}")  
 private String cupSize;  
  
 @RequestMapping(value = "/hello",method = RequestMethod.*GET*)  
 public String Say(){  
  
 return cupSize;  
 }  
}

一般会像下面的girl那样配置，然后通过一个类调用

**server:  
 port:** 8081  
 **context-path:** "/context"  
**girl:  
 cupSize:** B  
 **age:** 18  
 *#Content: "cupSize:${cupSize} age:${age}}"*

使用配置的类

package com.imooc.girl;  
  
import org.springframework.boot.context.properties.ConfigurationProperties;  
import org.springframework.stereotype.Component;  
  
@Component //想要被别的类引用需要加这行注解  
@ConfigurationProperties(prefix = "girl") //此名表示使用配置文件中的girl部分  
public class GirlProperties {  
  
 private String cupSize;  
 private int age;  
  
 public String getCupSize() {  
 return cupSize;  
 }  
  
 public void setCupSize(String cupSize) {  
 this.cupSize = cupSize;  
 }  
  
 public int getAge() {  
 return age;  
 }  
  
 public void setAge(int age) {  
 this.age = age;  
 }  
}

使用方法

package com.imooc.girl;  
  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.beans.factory.annotation.Value;  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RequestMethod;  
import org.springframework.web.bind.annotation.RestController;  
  
  
@RestController //用WEB的方式访问类  
public class HelloController {  
  
 @Autowired //使用配置类需要加这行注解  
 private GirlProperties girlproperties;  
  
  
 @RequestMapping(value = "/hello",method = RequestMethod.*GET*) //访问的目录，用GET的方式访问  
 public String Say(){  
  
 return girlproperties.getCupSize();  
 }  
}

使用多个配置文件

Application.yml

**spring:  
 profiles:  
 active:** dev

Application-dev.yml

**server:  
 port:** 8081  
 **context-path:** "/context"  
**girl:  
 cupSize:** B  
 **age:** 18  
 *#Content: "cupSize:${cupSize} age:${age}}"*

Application-prod.yml

**server:  
 port:** 8081  
 **context-path:** "/context"  
**girl:  
 cupSize:** F  
 **age:** 18  
 *#Content: "cupSize:${cupSize} age:${age}}"*

多个连接访问一个方法可以用下面的方法来设置

@RequestMapping(value = {"/hello","/hi"},method = RequestMethod.*GET*) //访问的目录，用GET的方式访问

@PathVariable使用连接中的参数(访问路径是http://localhost:8081/say/123)

@RequestMapping(value = "/say/{id}",method = RequestMethod.*GET*) //访问的目录，用GET的方式访问  
 public String Say(@PathVariable("id") int id){  
 return "ID:"+id;  
// return girlproperties.getCupSize();  
 }

或者参数在前面(http://localhost:8081/123/say)

@RequestMapping(value = "/{id}/say",method = RequestMethod.*GET*) //访问的目录，用GET的方式访问  
 public String Say(@PathVariable("id") int id){  
 return "ID:"+id;  
// return girlproperties.getCupSize();  
 }

@RequestParam可以使用?方式访问参数（如：<http://localhost:8081/say?id=111>）

@RequestMapping(value = "/say",method = RequestMethod.*GET*) //访问的目录，用GET的方式访问  
 public String Say(@RequestParam("id") int id){  
 return "ID:"+id;  
// return girlproperties.getCupSize();  
 }

默认值与是否必传

@RequestMapping(value = "/say",method = RequestMethod.*GET*) //访问的目录，用GET的方式访问  
 public String Say(@RequestParam(value = "id",required =false,defaultValue = "888") int id){  
 return "ID:"+id;  
// return girlproperties.getCupSize();  
 }

配置文件

**spring:  
 profiles:  
 active:** dev  
 **datasource:  
 driver-class-name:** com.mysql.jdbc.Driver  
 **url:** jdbc:mysql://localhost:3306/dbgirl?useSSL=false  
 **username:** sa  
 **password:** 801023  
 **jpa:  
 hibernate:  
 ddl-auto:** create  
 **show-sql:** true

数据库无需建表，只要建立类就可以对应

Service里负责处理验证

Controller里直接调用Service里的代码

ExceptionHandle来负责处理异常