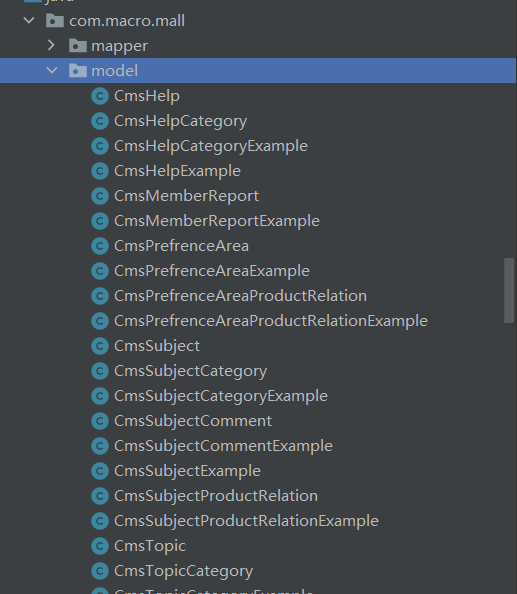
今日任务完成18

1. com.macro.mall.model





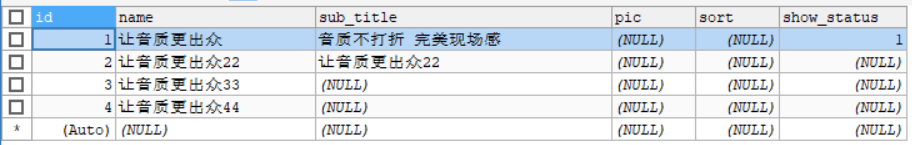
package com.macro.mall.model;  
  
import io.swagger.annotations.ApiModelProperty;  
import java.io.Serializable;  
  
public class CmsPrefrenceArea implements Serializable {

mall项目的实体类存放在(com.macro.mall.model)该包类，该CmsPrefrenceArea实体类属性有

private Long id;  
private String name;  
private String subTitle;  
private Integer sort;  
private Integer showStatus;

private byte[] pic;

如数据库表所示：



@ApiModelProperty(value = "展示图片")

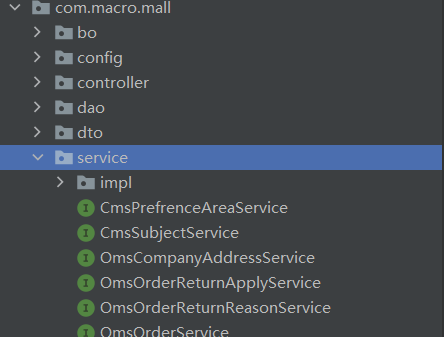
@ApiModelProperty()用于方法，字段； 表示对[model](https://so.csdn.net/so/search?q=model&spm=1001.2101.3001.7020)属性的说明或者数据操作更改  
value–字段说明  
name–[重写](https://so.csdn.net/so/search?q=%E9%87%8D%E5%86%99&spm=1001.2101.3001.7020)属性名字  
dataType–重写属性类型  
required–是否必填  
example–举例说明  
hidden–隐藏

private static final long *serialVersionUID* = 1L;

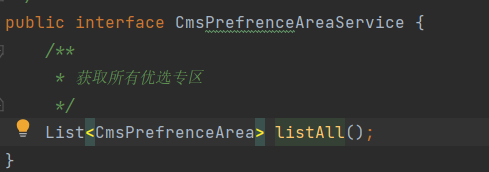
serialVersionUID适用于java序列化机制。简单来说，JAVA序列化的机制是通过 判断类的serialVersionUID来验证的版本一致的。在进行反序列化时，JVM会把传来的字节流中的serialVersionUID于本地相应实体类的serialVersionUID进行比较。如果相同说明是一致的，可以进行反序列化，否则会出现反序列化版本一致的异常，即是InvalidCastException。

**具体序列化的过程是这样的：**序列化操作时会把系统当前类的serialVersionUID写入到序列化文件中，当反序列化时系统会自动检测文件中的serialVersionUID，判断它是否与当前类中的serialVersionUID一致。如果一致说明序列化文件的版本与当前类的版本是一样的，可以反序列化成功，否则就失败

1. com.macro.mall.service







mall项目的该路径存放的是对应业务的接口。该图service层设计的是商品优选管理的业务接口，通过listAll方法将CmPrefrenceArea实体存取在list集合中

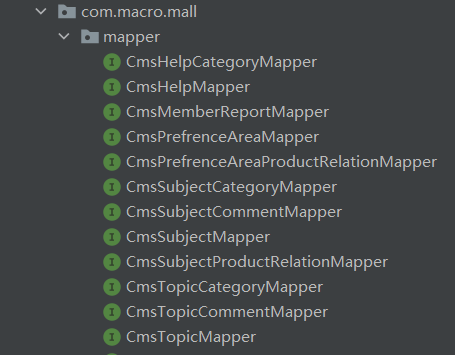
1. com.macro.mall.service.imp

此路径下存放是相应业务层接口的实现

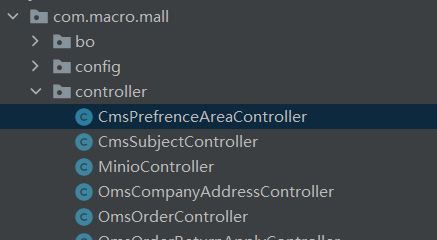


该实现类实现了对

4.com.macro.mall.mapper



5.com.macro.mall.controller



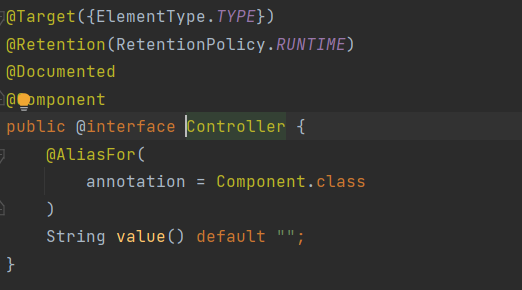
此路径存放的是负责具体的业务模块流程的控制controller ,controller层负责前后端交互，接受前端请求,调用service层返回的数据，最后返回具体的页面和数据到客户端。



@Controller

该注解类型用于指示Spring类的实例是一个控制器，其注解形式为@Controller。该注解在使用时不需要再实现Controller接口，只需要将@Controller注解加入到控制器类上，然后通过Spring的扫描机制找到标注了该注解的控制器即可。

@controller源码：



@Autowired

@Autowired是一种注解，可以对成员变量、方法和构造函数进行标注，来完成自动装配的工作，@Autowired标注可以放在成员变量上，也可以放在成员变量的set方法上，也可以放在任意方法上表示，自动执行当前方法，如果方法有参数，会在IOC容器中自动寻找同类型参数为其传值。

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

属性：value

指定请求的实际地址，指定的地址可以是URI Template 模式

@RequestMapping 中的 value 和 path 属性（这两个属性作用相同，可以互换，如果仅有这一个属性，则可以省略）

method

method 主要用来定义接收浏览器发来的何种请求。在Spring中，使用枚举类

org.springframework.web.bind.annotation.RequestMethod来定义浏览器请求的方式。

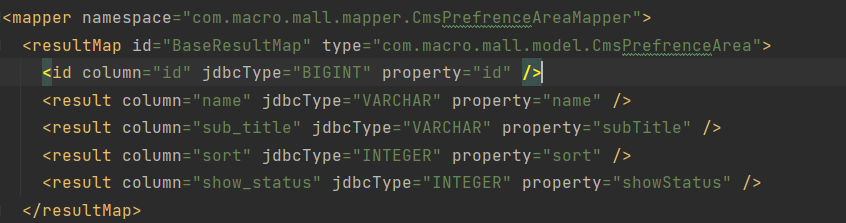
HTTP规范定义了多种请求资源的方式，最基本的有四种，分别为：GET（查）、POST（增）、PUT（改）、DELETE（删），而URL则用于定位网络上的资源相当于地址的作用，配合四种请求方式，可以实现对URL对应的资源的增删改查操作。

@ResponseBody

@responseBody注解的作用是将controller的方法返回的对象通过适当的转换器转换为指定的格式之后，写入到response对象的body区，通常用来返回JSON数据或者是XML数据。

这个注解表示该方法的返回结果直接写入HTTP response body中，一般在异步获取数据时使用。

在使用@RequestMapping后，返回值通常解析为跳转路径。加上@responsebody后，返回结果直接写入HTTP response body中，不会被解析为跳转路径。比如异步请求，希望响应的结果是json数据，那么加上@responsebody后，就会直接返回json数据。





Mybatis的mapper标签 namespace方法说明  
  
有三种全路径：  
namespace绑定实体类的全路径；  
namespace绑定dao层接口的全路径；  
namespace绑定mapper接口的sql.xml文件。  
  
第一种：namespace绑定**实体类**的全路径：  
当namespace绑定的是实体类的全路径时，其实现数据持久化的方式为无代理模式实现数据持久化，可以手动实现dao层的接口。  
<mapper namespace="com.system.entity.Book"></namespace>  
  
第二种：namespace绑定**dao层接口**的全路径：  
当namespace绑定的是dao接口的全路径时，其实现数据持久化的方式为有代理模式实现数据持久化，即会自动产生代理，自动实现数据的持久化，不需要实现dao层的接口。  
<mapper namespace="com.system.dao.LiveInfoDao"></namespace>  
  
第三种：namespace绑定**mapper接口**的sql.xml文件。  
当namespace绑定的是mapper接口对应的sql.xml文件时，其也是有代理模式自动实现数据持久化。  
<mapper namespace="com.system.mapper.BookMapper"></namespace>