

[spring&Springmvc&Springboo&mybatis]

[Spring&SpringMvc&SpringBoot&Mybatis学习文档]



2018-10-25

[公司名称]

[公司地址]

目录

[1 Spring 2](#_Toc528683046)

[1.1项目启动脚本及顺序（CentOS7、Eclipse） 2](#_Toc528683047)

[1.2项目启动顺序及端口 3](#_Toc528683048)

[1.3虚拟机web 3](#_Toc528683049)

[2 SpringMVC 3](#_Toc528683050)

[2.1为什么要使用Maven？ 3](#_Toc528683051)

[2.2后台工程搭建分析 4](#_Toc528683052)

[3 SpringBoot 4](#_Toc528683053)

[2.1为什么要使用Maven？ 4](#_Toc528683054)

[2.2后台工程搭建分析 4](#_Toc528683055)

[4 Mybatis 4](#_Toc528683056)

[4.1是什么 4](#_Toc528683057)

[4.2示例 4](#_Toc528683058)

[5 SSM 7](#_Toc528683059)

[2.1为什么要使用Maven？ 7](#_Toc528683060)

[4.2后台工程搭建分析 8](#_Toc528683061)

[6 SSM+ 8](#_Toc528683062)

[2.1为什么要使用Maven？ 8](#_Toc528683063)

[2.2后台工程搭建分析 8](#_Toc528683064)

[7 全局RestAPI统一返回 8](#_Toc528683065)

[7.1 BaseResult 8](#_Toc528683066)

[2.2 BasePageResult 11](#_Toc528683067)

[8 全局异常处理 13](#_Toc528683068)

[8.1 @RestControllerAdvice 13](#_Toc528683069)

[8.2异常拦截器 14](#_Toc528683070)

# 1 Spring

## 1.1项目启动脚本及顺序（CentOS7、Eclipse）

**虚拟机：**

**防火墙：**

systemctl status firewalld

systemctl start firewalld

**① systemctl stop firewalld**

**Zookeeper:**

cd /usr/local/zookeeper/zookeeper-3.4.6/bin/

./zkServer.sh stop

**② ./zkServer.sh start**

**Dubbo-Admin:**

cd /opt/apache-tomcat-7.0.59/bin/

**③ ./startup.sh**

**FastDFS:**

**④ /etc/init.d/fdfs\_trackerd start**

**⑤ /etc/init.d/fdfs\_storaged start**

**Nginx:**

**⑥ /usr/local/nginx/sbin/nginx**

**Redis:**

**单机版：**

cd /usr/local/redis/bin/

**⑦ ./redis-server redis.conf**

**集群版：**

cd /usr/local/redis-cluster/

**⑧ ./** **start-all.sh**

**⑨** **./redis-trib.rb create --replicas 1 192.168.188.128:7001 192.168.188.128:7002 192.168.188.128:7003 192.168.188.128:7004 192.168.188.128:7005 192.168.188.128:7006**

**ActiveMQ：**

**🔟** cd /opt/apache-activemq-5.12.0/bin

**./activemq start**

## 1.2项目启动顺序及端口

**项目：**

## 1.3虚拟机web

**Dubbo-admin:**

# 2 SpringMVC

## 2.1为什么要使用Maven？

## 2.2后台工程搭建分析

# 3 SpringBoot

## 2.1为什么要使用Maven？

## 2.2后台工程搭建分析

# 4 Mybatis

## 4.1是什么

MyBatis is a first class persistence framework with support for custom SQL, stored procedures and advanced mappings. MyBatis eliminates almost all of the JDBC code and manual setting of parameters and retrieval of results. MyBatis can use simple XML or Annotations for configuration and map primitives, Map interfaces and Java POJOs (Plain Old Java Objects) to database records.

## 4.2示例

UserMerchant selectUserMerchantByUserIdAndMerchantId(@Param(**"userId"**) **int** userId,@Param(**"merchantId"**) **int** merchantId);

*<!--根据用户id和商户id查询用户对应商户积分-->*<**select id="selectUserMerchantByUserIdAndMerchantId" resultMap="BaseResultMap"**>  
 select  
 <**include refid="Base\_Column\_List"**/>  
 from  
 user\_merchant um  
 <**where**>  
 <**if test="userId != null"**>  
 user\_id = #{userId,jdbcType=INTEGER}  
 </**if**>  
 <**if test="merchantId != null"**>  
 and merchant\_id = #{merchantId,jdbcType=INTEGER}  
 </**if**>  
 </**where**>  
</**select**>

PageList<UserMerchantByUserIdVO> getUserMerchantsByUserId(PageBounds pageBounds, UserMerchantByUserIdVO userMerchantByUserIdVO);

<**mapper namespace="com.zcy.zcmorefun.jifen.dao.UserMerchantMapper"** >  
 <**resultMap id="BaseResultMap" type="com.zcy.zcmorefun.jifen.model.system.UserMerchant"** >  
 <**id column="user\_merchant\_id" property="userMerchantId" jdbcType="INTEGER"** />  
 <**result column="user\_id" property="userId" jdbcType="INTEGER"** />  
 <**result column="user\_name" property="userName" jdbcType="VARCHAR"** />  
 <**result column="user\_phone\_number" property="userPhoneNumber" jdbcType="VARCHAR"** />  
 <**result column="user\_open\_id" property="userOpenId" jdbcType="VARCHAR"** />  
 <**result column="merchant\_id" property="merchantId" jdbcType="INTEGER"** />  
 <**result column="accredit\_time" property="accreditTime" jdbcType="TIMESTAMP"** />  
 <**result column="residue\_integral" property="residueIntegral" jdbcType="INTEGER"** />  
 <**result column="status" property="status" jdbcType="INTEGER"** />  
 </**resultMap**>  
  
 <**sql id="Base\_Column\_List"** >  
 user\_merchant\_id, user\_id, user\_name, user\_phone\_number, user\_open\_id, merchant\_id,  
 accredit\_time, residue\_integral, status  
 </**sql**>  
  
 <**resultMap id="UserMerchantByUserIdVOMap" type="com.zcy.zcmorefun.jifen.vo.UserMerchantByUserIdVO"** >  
 <**id column="user\_merchant\_id" property="userMerchantId" jdbcType="INTEGER"** />  
 <**result column="user\_id" property="userId" jdbcType="INTEGER"** />  
 <**result column="user\_name" property="userName" jdbcType="VARCHAR"** />  
 <**result column="user\_phone\_number" property="userPhoneNumber" jdbcType="VARCHAR"** />  
 <**result column="user\_open\_id" property="userOpenId" jdbcType="VARCHAR"** />  
 <**result column="merchant\_id" property="merchantId" jdbcType="INTEGER"** />  
 <**result column="accredit\_time" property="accreditTime" jdbcType="TIMESTAMP"** />  
 <**result column="residue\_integral" property="residueIntegral" jdbcType="INTEGER"** />  
 <**result column="status" property="status" jdbcType="INTEGER"** />  
 <**result column="merchant\_name" property="merchantName" jdbcType="VARCHAR"** />  
 <**result column="setup\_id" property="setupId" jdbcType="INTEGER"** />  
 <**result column="exchange\_rate" property="exchangeRate" jdbcType="INTEGER"** />  
 <**collection property="setupFixLimits" ofType="com.zcy.zcmorefun.jifen.model.system.SetupFixLimit"  
 select="com.zcy.zcmorefun.jifen.dao.SetupFixLimitMapper.selectBySetUpId" column="setup\_id"**></**collection**>  
 <**collection property="merchantInterfaces" ofType="com.zcy.zcmorefun.jifen.model.system.MerchantInterface"  
 select="com.zcy.zcmorefun.jifen.dao.MerchantInterfaceMapper.selectByMerchantId" column="merchant\_id"**></**collection**>  
 </**resultMap**>  
  
 *<!--可绑定但是未绑定的商户-->* <**resultMap id="UnRegisteredVOMap" type="com.zcy.zcmorefun.jifen.vo.UserMerchantByUserIdVO"** >  
 <**id column="user\_merchant\_id" property="userMerchantId" jdbcType="INTEGER"** />  
 <**result column="user\_id" property="userId" jdbcType="INTEGER"** />  
 <**result column="user\_name" property="userName" jdbcType="VARCHAR"** />  
 <**result column="user\_phone\_number" property="userPhoneNumber" jdbcType="VARCHAR"** />  
 <**result column="user\_open\_id" property="userOpenId" jdbcType="VARCHAR"** />  
 <**result column="merchant\_id" property="merchantId" jdbcType="INTEGER"** />  
 <**result column="accredit\_time" property="accreditTime" jdbcType="TIMESTAMP"** />  
 <**result column="residue\_integral" property="residueIntegral" jdbcType="INTEGER"** />  
 <**result column="status" property="status" jdbcType="INTEGER"** />  
 <**result column="merchant\_name" property="merchantName" jdbcType="VARCHAR"** />  
 <**result column="setup\_id" property="setupId" jdbcType="INTEGER"** />  
 <**result column="exchange\_rate" property="exchangeRate" jdbcType="INTEGER"** />  
 <**collection property="merchantInterfaces" ofType="com.zcy.zcmorefun.jifen.model.system.MerchantInterface"  
 select="com.zcy.zcmorefun.jifen.dao.MerchantInterfaceMapper.selectByMerchantId" column="merchant\_id"**></**collection**>  
 </**resultMap**>

*<!--根据用户id查询用户已经注册商户-->* <**select id="getUserMerchantsByUserId" resultMap="UserMerchantByUserIdVOMap" parameterType="com.zcy.zcmorefun.jifen.vo.UserMerchantByUserIdVO"**>  
 select  
 u.user\_merchant\_id,  
 u.user\_id,  
 u.user\_name,  
 u.user\_phone\_number,  
 u.user\_open\_id,  
 u.merchant\_id,  
 u.accredit\_time,  
 u.residue\_integral,  
 u.status,  
 m.merchant\_name,  
 ms.setup\_id  
 from  
 user\_merchant u  
 left join merchant\_info m on u.merchant\_id=m.merchant\_id left join merchant\_setup ms on m.merchant\_id=ms.merchant\_id  
 <**where**>  
 <**if test="userId != null"**>  
 user\_id = #{userId,jdbcType=INTEGER}  
 </**if**>  
 <**if test="status != null"**>  
 and status = #{status,jdbcType=INTEGER}  
 </**if**>  
 </**where**>  
 order by u.accredit\_time desc  
 </**select**>

**void** insertAll(List<MerchantInterfaceVO> merchantInterfaceVOS);

*<!--商户接口批量插入-->*<**insert id="insertAll" parameterType="java.util.List" useGeneratedKeys="true" keyProperty="interfaceId"**>  
 INSERT INTO merchant\_interface (interface\_id, merchant\_id, interface\_name,  
 interface\_url, interface\_desc, interface\_status,  
 interface\_type, create\_time)  
 VALUES  
 <**foreach collection="list" item="item" index="index" separator=","**>  
 (#{item.interfaceId},#{item.merchantId},#{item.interfaceName},#{item.interfaceUrl},#{item.interfaceDesc},#{item.interfaceStatus},#{item.interfaceType},#{item.createTime})  
 </**foreach**>  
</**insert**>

# 5 SSM

## 2.1为什么要使用Maven？

## 4.2后台工程搭建分析

# 6 SSM+

## 2.1为什么要使用Maven？

## 2.2后台工程搭建分析

# 7 全局RestAPI统一返回

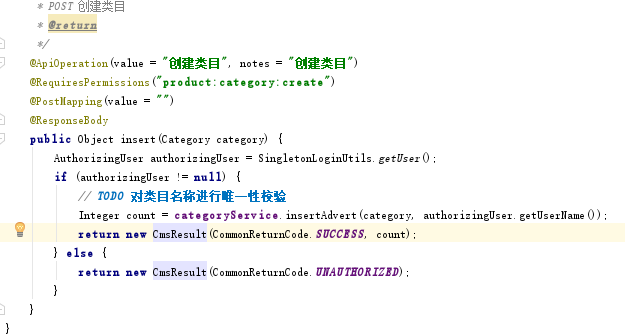
## 7.1 BaseResult

**public class** BaseResult **implements** Serializable{  
   
 **private static final long *serialVersionUID*** = 1L;  
  
 */\*\* 返回状态码 \*/* **private** Integer **code**;  
   
 */\*\* 返回信息 \*/* **private** String **message**;  
   
 */\*\* 返回数据 \*/* **private** Object **data**;  
   
 **public** BaseResult(Integer code, String message) {  
 **super**();  
 **this**.**code** = code;  
 **this**.**message** = message;  
 }  
  
 **public** BaseResult(Integer code, String message, Object data) {  
 **super**();  
 **this**.**code** = code;  
 **this**.**message** = message;  
 **this**.**data** = data;  
 }  
  
 **public** Integer getCode() {  
 **return code**;  
 }  
  
 **public void** setCode(Integer code) {  
 **this**.**code** = code;  
 }  
  
 **public** String getMessage() {  
 **return message**;  
 }  
  
 **public void** setMessage(String message) {  
 **this**.**message** = message;  
 }  
  
 **public** Object getData() {  
 **return data**;  
 }  
  
 **public void** setData(Object data) {  
 **this**.**data** = data;  
 }  
}

**public class** CmsResult **extends** BaseResult {  
  
 **private static final long *serialVersionUID*** = 1L;  
  
 **public** CmsResult(ReturnCode returnCode) {  
 **super**(returnCode.getCode(), returnCode.getMessage());  
 }  
   
 **public** CmsResult(ReturnCode returnCode, Object data) {  
 **super**(returnCode.getCode(), returnCode.getMessage(), data);  
 }  
   
 **public** CmsResult(Integer code, String message) {  
 **super**(code, message);  
 }  
}

**public interface** ReturnCode {  
   
 **public** Integer getCode();  
  
 **public** String getMessage();  
   
}

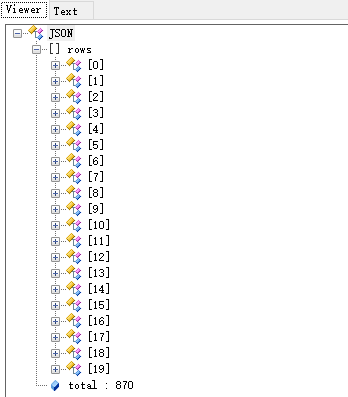
**public enum** CommonReturnCode **implements** ReturnCode {  
   
 */\*\* 请求失败 \*/* ***FAILED***(0, **"failed"**),  
 */\*\* 请求成功 \*/* ***SUCCESS***(1, **"success"**),  
 */\*\* 未知错误 \*/* ***UNKNOWN\_ERROR***(-1, **"未知错误,请联系管理员!"**),  
   
 */\*\* 200请求成功 \*/* ***OK***(200, **"请求成功"**),  
 */\*\* 207频繁操作 \*/* ***MULTI\_STATUS***(207, **"频繁操作"**),  
   
 */\*\* 303登录失败 \*/* ***LOGIN\_FAIL***(303, **"登录失败 "**),  
   
 */\*\* 400请求参数出错 \*/* ***BAD\_REQUEST***(400, **"请求参数出错"**),  
 */\*\* 401没有登录 \*/* ***UNAUTHORIZED***(401, **"您未登录或者登录已超时,请先登录!"**),  
 */\*\* 403没有权限 \*/* ***FORBIDDEN***(403, **"没有权限"**),  
 */\*\* 404找不到页面 \*/* ***NOT\_FOUND***(404, **"找不到页面"**),  
 */\*\* 408请求超时 \*/* ***REQUEST\_TIMEOUT***(408, **"请求超时"**),  
 */\*\* 409发生冲突 \*/* ***CONFLICT***(409, **"发生冲突"**),  
 */\*\* 410已被删除 \*/* ***GONE***(410, **"已被删除"**),  
 */\*\* 423已被锁定 \*/* ***LOCKED***(423, **"已被锁定"**),  
   
 */\*\* 500服务器出错 \*/* ***INTERNAL\_SERVER\_ERROR***(500, **"服务器出错"**);  
   
 */\*\* 返回状态码 \*/* **private** Integer **code**;  
  
 */\*\* 返回消息 \*/* **private** String **message**;  
   
 **private** CommonReturnCode(Integer code, String message) {  
 **this**.**code** = code;  
 **this**.**message** = message;  
 }  
  
 @Override  
 **public** Integer getCode() {  
 **return code**;  
 }  
  
 @Override  
 **public** String getMessage() {  
 **return message**;  
 }  
}



## 2.2 BasePageResult

**public class** CmsPageResult **implements** Serializable{  
   
 **private static final long *serialVersionUID*** = 1L;  
   
 **private** Object **rows**;  
   
 **private** Integer **total**;  
   
 **public** CmsPageResult(Object rows, Integer total) {  
 **super**();  
 **this**.**rows** = rows;  
 **this**.**total** = total;  
 }  
  
 **public** Object getRows() {  
 **return rows**;  
 }  
  
 **public void** setRows(Object rows) {  
 **this**.**rows** = rows;  
 }  
  
 **public** Integer getTotal() {  
 **return total**;  
 }  
  
 **public void** setTotal(Integer total) {  
 **this**.**total** = total;  
 }  
}

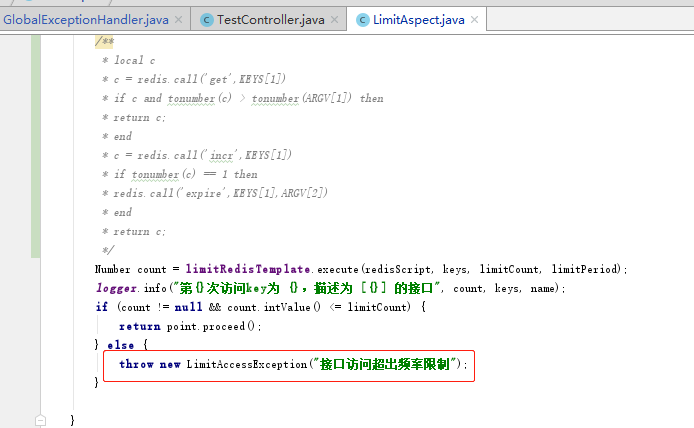




# 8 全局异常处理

## 8.1 @RestControllerAdvice

@RestControllerAdvice  
@Order(value = Ordered.***HIGHEST\_PRECEDENCE***)  
**public class** GlobalExceptionHandler {  
  
 @ExceptionHandler(value = AuthorizationException.**class**)  
 **public** Object handleAuthorizationException(HttpServletRequest request) {  
 **if** (*isAjaxRequest*(request)) {  
 **return** ResponseBo.*error*(**"暂无权限，请联系管理员！"**);  
 } **else** {  
 ModelAndView mav = **new** ModelAndView();  
 mav.setViewName(**"error/403"**);  
 **return** mav;  
 }  
 }  
  
 @ExceptionHandler(value = ExpiredSessionException.**class**)  
 **public** String handleExpiredSessionException() {  
 **return "login"**;  
 }  
  
 @ExceptionHandler(value = LimitAccessException.**class**)  
 **public** ResponseBo handleLimitAccessException(LimitAccessException e) {  
 **return** ResponseBo.*error*(e.getMessage());  
 }  
  
 @ExceptionHandler(value = ArithmeticException.**class**)  
 **public** ResponseBo testArithmeticException(ArithmeticException e) {  
 **return** ResponseBo.*error*(e.getMessage());  
 }  
  
 **private static boolean** isAjaxRequest(HttpServletRequest request) {  
 **return** (request.getHeader(**"X-Requested-With"**) != **null** && **"XMLHttpRequest"**.equals(request.getHeader(**"X-Requested-With"**)));  
 }  
}



## 8.2异常拦截器