1. 工具类和实体类
2. 在utils包下创建工具类NumberUtil，该工具类用于生成订单号
3. 在entity包下创建如下po实体类
4. 订单实体类：QstMallOrder
5. 订单项实体类：QstMallOrderItem
6. 在com.qst.qstmall.controller.VO包下创建如下VO实体类
7. 订单详情VO：QstMallOrderDetailVO （订单详情页展示的数据）
8. 订单列表VO：QstMallOrderListVO （订单列表页面展示的数据）
9. 订单项VO：QstMallOrderItemVO 订单中订单项（商品）
10. 在common包下
11. 创建保存订单状态的枚举类：PayStatusEnum
12. 创建保存支付类型的枚举类：PayTypeEnum
13. 创建保存订单状态的枚举类：QstMallOrderStatusEnum
14. Dao层实现
15. dao包中
16. 新建接口QstMallOrderItemMapper：声明订单项操作相关方法
17. 新建接口QstMallOrderMapper：声明订单操作相关方法的接口
18. 在已有的MallUserMapper中新增如下两个方法的声明  
    int updateByPrimaryKeySelective(MallUser record);  
    MallUser selectByPrimaryKey(Long userId);
19. 在mapper目录下
20. 创建QstMallOrderItemMapper.xml
21. 创建QstMallOrderMapper.xml
22. 在已有的MallUserMapper.xml文件新增如下实现

<update id="updateByPrimaryKeySelective" parameterType="com.qst.qstmall.entity.MallUser">  
 update tb\_qst\_mall\_user  
 <set>  
 <if test="nickName != null">  
 nick\_name = #{nickName,jdbcType=VARCHAR},  
 </if>  
 <if test="loginName != null">  
 login\_name = #{loginName,jdbcType=VARCHAR},  
 </if>  
 <if test="passwordMd5 != null">  
 password\_md5 = #{passwordMd5,jdbcType=VARCHAR},  
 </if>  
 <if test="introduceSign != null">  
 introduce\_sign = #{introduceSign,jdbcType=VARCHAR},  
 </if>  
 <if test="address != null">  
 address = #{address,jdbcType=VARCHAR},  
 </if>  
 <if test="isDeleted != null">  
 is\_deleted = #{isDeleted,jdbcType=TINYINT},  
 </if>  
 <if test="lockedFlag != null">  
 locked\_flag = #{lockedFlag,jdbcType=TINYINT},  
 </if>  
 <if test="createTime != null">  
 create\_time = #{createTime,jdbcType=TIMESTAMP},  
 </if>  
 </set>  
 where user\_id = #{userId,jdbcType=BIGINT}  
</update>  
  
  
<select id="selectByPrimaryKey" parameterType="java.lang.Long" resultMap="BaseResultMap">  
 select  
 <include refid="Base\_Column\_List"/>  
 from tb\_qst\_mall\_user  
 where user\_id = #{userId,jdbcType=BIGINT} and is\_deleted = 0  
</select>

1. Service层实现
2. 在service包中
3. 新建QstMallOrderService接口封装订单操作
4. 在已有的QstMallUserService中新增如下方法声明

QstMallUserVO updateUserInfo(MallUser mallUser, HttpSession httpSession);

1. 在service.impl包中
2. 新建QstMallOrderServiceImpl类，实现QstMallOrderService接口
3. 在已有的QstMallUserServiceImpl中新增如下方法实现，该方法用来修改用户信息

@Override  
public QstMallUserVO updateUserInfo(MallUser mallUser, HttpSession httpSession) {  
 MallUser user = mallUserMapper.selectByPrimaryKey(mallUser.getUserId());  
 if (user != null) {  
 user.setNickName(mallUser.getNickName());  
 user.setAddress(mallUser.getAddress());  
 user.setIntroduceSign(mallUser.getIntroduceSign());  
 if (mallUserMapper.updateByPrimaryKeySelective(user) > 0) {  
 QstMallUserVO qstMallUserVO = new QstMallUserVO();  
 user = mallUserMapper.selectByPrimaryKey(mallUser.getUserId());  
 BeanUtil.*copyProperties*(user, qstMallUserVO);  
 httpSession.setAttribute(Constants.*MALL\_USER\_SESSION\_KEY*, qstMallUserVO);  
 return qstMallUserVO;  
 }  
 }  
 return null;  
}

1. Controller层实现
2. 在controller包中新建OrderController用于处理订单相关请求
3. 在已有的PersonalController.java中新增如下方法，处理个人中心相关请求

@PostMapping("/personal/updateInfo")  
@ResponseBody  
public Result updateInfo(@RequestBody MallUser mallUser, HttpSession httpSession) {  
 QstMallUserVO mallUserTemp = qstMallUserService.updateUserInfo(mallUser,httpSession);  
 if (mallUserTemp == null) {  
 Result result = ResultGenerator.*genFailResult*("修改失败");  
 return result;  
 } else {  
 *//返回成功* Result result = ResultGenerator.*genSuccessResult*();  
 return result;  
 }  
}  
  
*/\*退出登录\*/*@GetMapping("/logout")  
public String logout(HttpSession httpSession) {  
 httpSession.removeAttribute(Constants.*MALL\_USER\_SESSION\_KEY*);  
 return "mall/login";  
}  
*/\*个人信息请求\*/*@GetMapping("/personal")  
public String personalPage(HttpServletRequest request) {  
 request.setAttribute("path", "personal");  
 return "mall/personal";  
}

1. 视图层实现

在resources->templates->mall目录下创建如下html文件

1. alipay.html：支付宝支付页面
2. my-orders.html:个人中心我的订单页面
3. order-detail.html:订单详情页
4. order-settle.html:订单确定页面
5. personal.html：个人中心
6. personal-sidebar.html：个人中心共用左侧导航栏
7. wxpay.html:微信支付页面
8. pay-select.html:支付方式选择页面