接口定义规则

- 创建表记录：POST /{控制器路由名称}/create
- 修改表记录：POST /{控制器路由名称}/update/{id}
- 删除指定表记录：POST /{控制器路由名称}/delete/{id}
- 分页查询表记录：GET /{控制器路由名称}/list
- 获取指定记录详情：GET /{控制器路由名称}/{id}

商品模块数据库表解析（一）

本文主要对商品分类、品牌管理、商品类型这三个功能的表进行解析，采用功能与表结构对照的形式。表解析只会标注一些需要理解的字段，简单字段请自行对照表注释。

商品分类

商品分类表

```
create table pms_product_category
(
 id bigint not null auto_increment,
 parent_id bigint comment '上级分类的编号：0表示一级分类',
 name varchar(64) comment '名称',
 level int(1) comment '分类级别：0->1级；1->2级',
 product_count int comment '商品数量',
 product_unit varchar(64) comment '商品单位',
 nav_status int(1) comment '是否显示在导航栏：0->不显示；1->显示',
 show_status int(1) comment '显示状态：0->不显示；1->显示',
 sort int comment '排序',
 icon varchar(255) comment '图标',
 keywords varchar(255) comment '关键字',
 description text comment '描述',
 primary key (id)
);
```

管理端展现

- 商品分类列表

- 添加商品分类

1. 初始化级联下拉列表数据
2. 保存相应数据
3. 编辑状态--初始化以保存筛选属性

移动端展现

品牌管理

商品品牌表

```
create table pms_brand
(
 id bigint not null auto_increment,
 name varchar(64) comment '名称',
 first_letter varchar(8) comment '首字母',
 sort int comment '排序',
 factory_status int(1) comment '是否为品牌制造商：0->不是；1->是',
 show_status int(1) comment '是否显示',
 product_count int comment '产品数量',
 product_comment_count int comment '产品评论数量',
 logo varchar(255) comment '品牌logo',
 big_pic varchar(255) comment '专区大图',
 brand_story text comment '品牌故事',
 primary key (id)
);
```

管理端展现

- 品牌列表

数据列表

添加

编号	品牌名称	品牌首字母	排序	品牌制造商	是否显示	相关	操作
6	小米	M	500			商品： 100 评价： 1000	编辑 删除
49	七匹狼	S	200			商品： 100 评价： 1000	编辑 删除
50	海澜之家	H	200			商品： 100 评价： 1000	编辑 删除
51	苹果	A	200			商品： 100 评价： 1000	编辑 删除
2	三星	S	100			商品： 100 评价： 1000	编辑 删除
3	华为	H	100			商品： 100 评价： 1000	编辑 删除
4	格力	G	30			商品： 100 评价： 1000	编辑 删除
5	方太	F	20			商品： 100 评价： 1000	编辑 删除
1	万和	W	0			商品： 100 评价： 1000	编辑 删除
21	OPPO	O	0			商品： 100 评价： 1000	编辑 删除

- 添加品牌

移动端展现

商品类型

商品类型即商品属性，主要是指商品的规格和参数，规格用于用户购买商品时选择，参数用于标示商品属性及搜索时筛选。

相关表结构

商品类型表

```
create table pms_product_attribute_category
(
 id bigint not null auto_increment,
 name varchar(64) comment '名称',
 attribute_count int comment '属性数量',
 param_count int comment '参数数量',
 primary key (id)
);
```

视频类型—商品属性表

type字段用于控制其是规格还是参数

```
create table pms_product_attribute
(
 id bigint not null auto_increment,
 product_attribute_category_id bigint comment '商品属性分类id',
 name varchar(64) comment '名称',
 select_type int(1) comment '属性选择类型：0->唯一；1->单选；2->多选；对应属性和参数意义不同；',
 input_type int(1) comment '属性录入方式：0->手工录入；1->从列表中选取',
 input_list varchar(255) comment '可选值列表，以逗号隔开',
 sort int comment '排序字段：最高的可以单独上传图片',
 filter_type int(1) comment '分类筛选样式：1->普通；1->颜色',
 search_type int(1) comment '检索类型：0->不需要进行检索；1->关键字检索；2->范围检索',
 related_status int(1) comment '相同属性产品是否关联：0->不关联；1->关联',
 hand_add_status int(1) comment '是否支持手动新增：0->不支持；1->支持',
 type int(1) comment '属性的类型：0->规格；1->参数',
 primary key (id)
);
```

商品分类和属性的关系表

用于选中分类后搜索时生成筛选属性。

```sql
create table pms_product_category_attribute_relation
(
 id bigint not null auto_increment,
 product_category_id bigint comment '商品分类id',
 product_attribute_id bigint comment '商品属性id',
 primary key (id)
);
```

管理端展现

- 商品属性分类列表

- 添加商品属性分类

- 商品规格列表

- 商品参数列表

数据列表

编号	属性名称	商品类型	属性是否可选	属性值的录入方式	可选值列表	排序
45	屏幕尺寸	手机数码-手机通讯	唯一	手工录入		0
46	网络	手机数码-手机通讯	唯一	从列表中选取	3G,4G	0
47	系统	手机数码-手机通讯	唯一	从列表中选取	Android,IOS	0
48	电池容量	手机数码-手机通讯	唯一	手工录入		0

批量操作 确定 共 4 条 5条/页 1

- 添加商品属性

- 添加商品时，选中商品属性分类，就会显示该分类的属性，用于生成sku

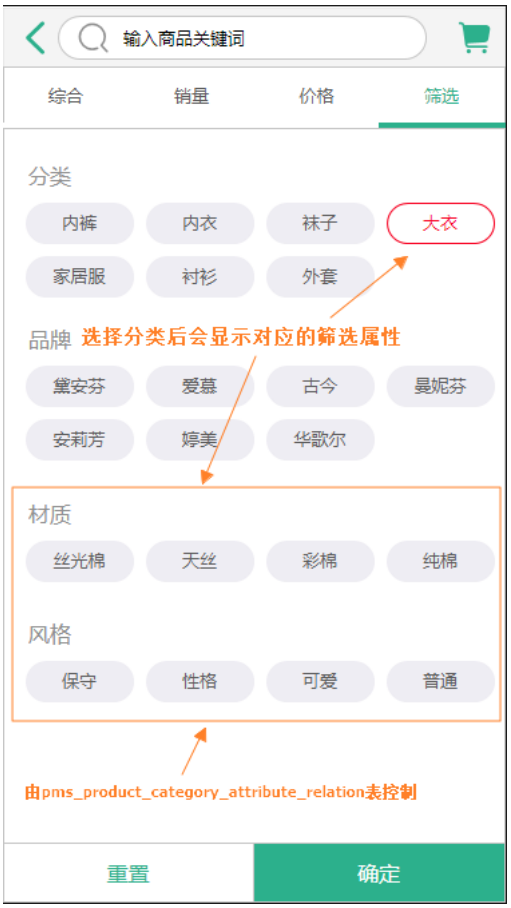
- 添加商品时，选中商品属性分类，会显示该分类的参数用于录入

移动端展现

- 选择商品规格

- 查看商品参数

- 搜索商品时用于选择分类后的筛选



商品管理

相关表结构

- **商品表**

商品信息主要包括四部分：商品的基本信息、商品的促销信息、商品的属性信息、商品的关联，商品表是整个商品的基本信息部分。

```
create table pms_product
(
 id bigint not null auto_increment,
 brand_id bigint comment '品牌id',
 product_category_id bigint comment '商品分类id',
 feight_template_id bigint comment '运费模版id',
 product_attribute_category_id bigint comment '商品属性分类id',
 name varchar(64) not null comment '商品名称',
 pic varchar(255) comment '图片',
 product_sn varchar(64) not null comment '货号',
 delete_status int(1) comment '删除状态：0->未删除；1->已删除',
 publish_status int(1) comment '上架状态：0->下架；1->上架',
 new_status int(1) comment '新品状态:0->不是新品；1->新品',
 recommand_status int(1) comment '推荐状态；0->不推荐；1->推荐',
 verify_status int(1) comment '审核状态：0->未审核；1->审核通过',
 sort int comment '排序',
 sale int comment '销量',
 price decimal(10,2) comment '价格',
 promotion_price decimal(10,2) comment '促销价格',
 gift_growth int default 0 comment '赠送的成长值',
 gift_point int default 0 comment '赠送的积分',
 use_point_limit int comment '限制使用的积分数',
 sub_title varchar(255) comment '副标题',
 description text comment '商品描述',
```

```
 original_price decimal(10,2) comment '市场价',
 stock int comment '库存',
 low_stock int comment '库存预警值',
 unit varchar(16) comment '单位',
 weight decimal(10,2) comment '商品重量，默认为克',
 preview_status int(1) comment '是否为预告商品：0->不是；1->是',
 service_ids varchar(64) comment '以逗号分割的产品服务：1->无忧退货；2->快速退款；3->免费包邮',
 keywords varchar(255) comment '关键字',
 note varchar(255) comment '备注',
 album_pics varchar(255) comment '画册图片，连产品图片限制为5张，以逗号分割',
 detail_title varchar(255) comment '详情标题',
 detail_desc text comment '详情描述',
 detail_html text comment '产品详情网页内容',
 detail_mobile_html text comment '移动端网页详情',
 promotion_start_time datetime comment '促销开始时间',
 promotion_end_time datetime comment '促销结束时间',
 promotion_per_limit int comment '活动限购数量',
 promotion_type int(1) comment '促销类型：0->没有促销使用原价;1->使用促销价；2->使用会员价；3->使用阶梯价格；4->使用满减价格；5->限时购',
 product_category_name varchar(255) comment '产品分类名称',
 brand_name varchar(255) comment '品牌名称',
 primary key (id)
);
```

```
eyJhbGciOiJIUzUxMiJ9.
eyJ1c2VyX25hbWUiOiJ0ZXN0IiwiY3JlYXRlZCI6MTYxNjEzMjg4MjU2NywiZXhwIjoxNjE2NzM3NjgyfQ.
_U2TYVPoSa4YjiOw5FxIaubCYWmNXJtaSKSmUqaBriA1qucCTEp9TGaDvjRcXbDZ04sJ8Yow5efdqbnnszaLTQ
```

- **商品SKU表**

SKU(Stock Keeping Unit)是指库存量单位，SPU(Standard Product Unit)是指标准产品单位。举个例子：iphone xs是一个SPU，而iphone xs 公开版 64G 银色是一个SKU。

```
create table pms_sku_stock
(
 id bigint not null auto_increment,
 product_id bigint comment '商品id',
 sku_code varchar(64) not null comment 'sku编码',
 price decimal(10,2) comment '价格',
 stock int default 0 comment '库存',
 low_stock int comment '预警库存',
 sp1 varchar(64) comment '规格属性1',
 sp2 varchar(64) comment '规格属性2',
 sp3 varchar(64) comment '规格属性3',
 pic varchar(255) comment '展示图片',
 sale int comment '销量',
 promotion_price decimal(10,2) comment '单品促销价格',
 lock_stock int default 0 comment '锁定库存',
 primary key (id)
);
```

- **商品阶梯价格表**

商品优惠相关表，购买同商品满足一定数量后，可以使用打折价格进行购买。如：买两件商品可以打八折。

```
create table pms_product_ladder
(
 id bigint not null auto_increment,
 product_id bigint comment '商品id',
 count int comment '满足的商品数量',
 discount decimal(10,2) comment '折扣',
 price decimal(10,2) comment '折后价格',
 primary key (id)
);
```

- **商品满减表**

商品优惠相关表，购买同商品满足一定金额后，可以减免一定金额。如：买满1000减100元。

```
create table pms_product_full_reduction
(
 id bigint not null auto_increment,
 product_id bigint comment '商品id',
 full_price decimal(10,2) comment '商品满足金额',
 reduce_price decimal(10,2) comment '商品减少金额',
 primary key (id)
);
```

- **商品会员价格表**

根据不同会员等级，可以以不同的会员价格购买。此处设计有缺陷，可以做成不同会员等级可以减免多少元或者按多少折扣进行购买。

```
create table pms_member_price
(
 id bigint not null auto_increment,
 product_id bigint comment '商品id',
 member_level_id bigint comment '会员等级id',
 member_price decimal(10,2) comment '会员价格',
 member_level_name varchar(100) comment '会员等级名称',
 primary key (id)
);
```

商品属性值表

如果对应的参数是规格且规格支持手动添加，那么该表用于存储手动新增的值；如果对应的商品属性是参数，那么该表用于存储参数的值。

```
create table pms_product_attribute_value
(
 id bigint not null auto_increment,
 product_id bigint comment '商品id',
 product_attribute_id bigint comment '商品属性id',
 value varchar(64) comment '手动添加规格或参数的值，参数单值，规格有多个时以逗号隔开',
 primary key (id)
);
```

后台展现

填写商品信息


1 填写商品信息
2 填写商品促销
3 填写商品属性
4 选择商品关联
* 商品分类: 手机数码 / 手机通讯
product_category_id
* 商品名称: 华为 HUAWEI P20
* 副标题: AI智慧全面屏 6GB +64GB 亮黑色 全网通版 移动联通电信4G手机 双卡双待手
* 商品品牌: 华为
brand_id
商品介绍: 请输入内容
商品货号: 6946605
商品售价: 3788
市场价: 4288
商品库存: 1000
计量单位: 件
商品重量: 0 克
排序 100
下一步，填写商品促销

填写商品促销


填写商品信息
2 填写商品促销
3 填写商品属性
4 选择商品关联
赠送积分: 3788
赠送成长值: 3788
积分购买限制: 0
预告商品:
preview_status
商品上架:
publish_status
商品推荐: 新品 推荐
服务保证: 无忧退货 快速退款 免费包邮
service_ids
详细页标题:
详细页描述:
商品关键字:
商品备注:
选择优惠方式: 无优惠 特惠促销 会员价格 阶梯价格 满减价格
promotion_type
上一步，填写商品信息
下一步，填写商品属性

特惠促销

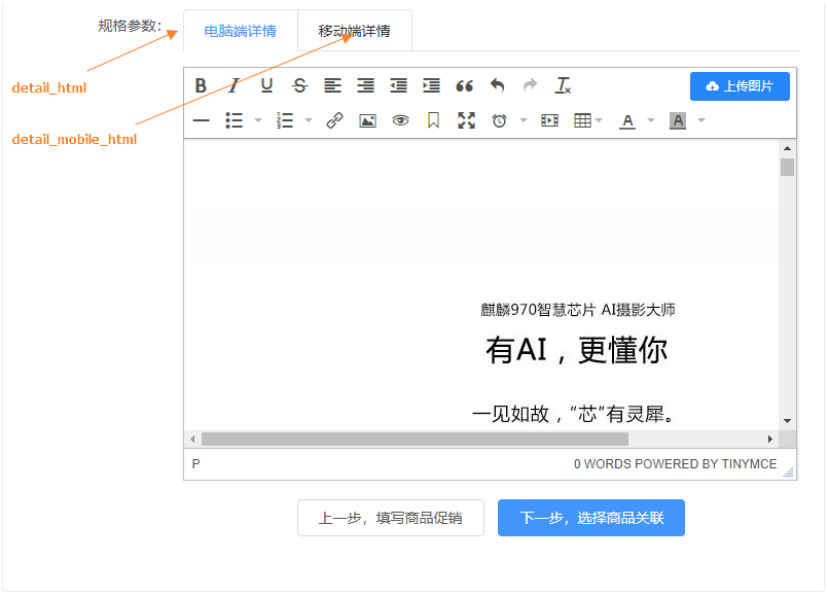
会员价格

阶梯价格

满减价格

填写商品属性


填写商品信息
填写商品促销
3
填写商品属性
4
选择商品关联
属性类型:
手机数码-手机通讯
product_attribute_category_id
商品规格:
颜色:
金色 删除
银色 删除
增加
容量:
16G 32G 64G 128G
根据选中的商品属性分类获取
pms_sku_stock表
颜色 容量 销售价格 商品库存 库存预警值 SKU编号 操作
金色 16G 3788 499 20180(删除
金色 32G 3999 500 20180(删除
银色 16G 3788 500 20180(删除
银色 32G 3999 500 20180(删除
刷新列表
同步价格
属性图片:
点击上传
金色: 只能上传jpg/png文件，且不超过10MB
点击上传
银色: 只能上传jpg/png文件，且不超过10MB
pms_sku_stock表中的pic
根据选中的商品属性分类来展示
商品参数:
屏幕尺寸: 5.0
网络: 4G
系统: Android
电池容量: 3000
pms_product_attribute_va
商品相册:
pms_product表中album_pics



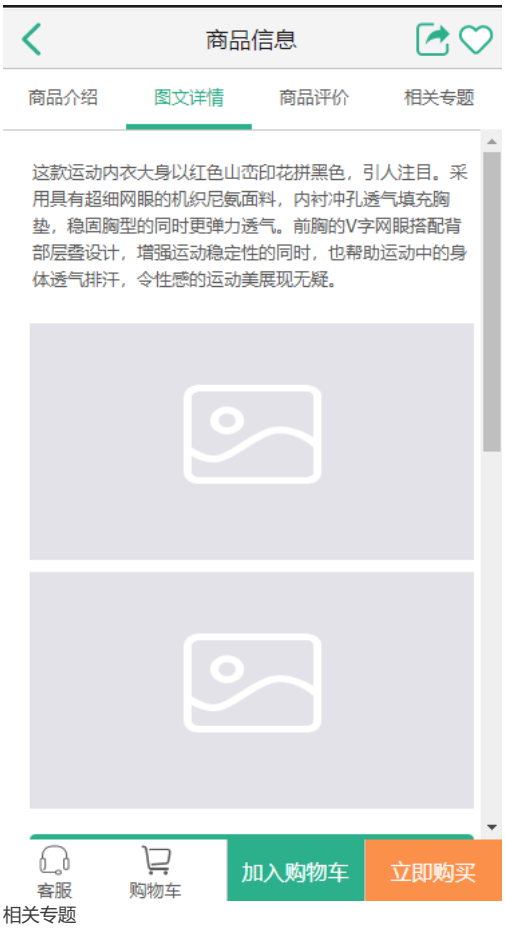
选择商品关联

移动端展现

商品介绍


商品信息
商品介绍
图文详情
商品评价
相关专题
1/4
银色星芒刺绣网纱底裤
薄如蝉翼，丝滑如肌肤
￥99 ￥199
秒杀抢购中
本场结束剩余 00 20 56
服务说明 无忧退货 快速退款 免费包邮
商品参数 查看
选择规格 肤色/M
商品介绍
这款运动内衣大身以红色山峦印花拼黑色，引人注目。采用具有超细网眼的机织尼氨面料，内衬冲孔透气填充胸垫，稳固胸型的同时更弹力透气。前胸的V字网眼搭配背
客服
购物车
加入购物车
立即购买

图文详情


商品信息
商品介绍
图文详情
商品评价
相关专题
这款运动内衣大身以红色山峦印花拼黑色，引人注目。采用具有超细网眼的机织尼氨面料，内衬冲孔透气填充胸垫，稳固胸型的同时更弹力透气。前胸的V字网眼搭配背部层叠设计，增强运动稳定性的同时，也帮助运动中的身体透气排汗，令性感的运动美展现无疑。
客服
购物车
加入购物车
立即购买

相关专题

商品评价及回复

相关表结构

商品评价表

```
create table pms_comment
(
 id bigint not null auto_increment,
 product_id bigint comment '商品id',
 member_nick_name varchar(255) comment '会员昵称',
 product_name varchar(255) comment '商品名称',
 star int(3) comment '评价星数：0->5',
 member_ip varchar(64) comment '评价的ip',
 create_time datetime comment '创建时间',
 show_status int(1) comment '是否显示',
 product_attribute varchar(255) comment '购买时的商品属性',
 collect_couont int comment '收藏数',
 read_count int comment '阅读数',
 content text comment '内容',
 pics varchar(1000) comment '上传图片地址，以逗号隔开',
 member_icon varchar(255) comment '评论用户头像',
 replay_count int comment '回复数',
 primary key (id)
);
```

产品评价回复表

```
create table pms_comment_replay
(
 id bigint not null auto_increment,
 comment_id bigint comment '评论id',
 member_nick_name varchar(255) comment '会员昵称',
 member_icon varchar(255) comment '会员头像',
 content varchar(1000) comment '内容',
 create_time datetime comment '创建时间',
 type int(1) comment '评论人员类型；0->会员；1->管理员',
```

```
10  primary key (id)
11 );
```

移动端展现

商品评价列表

商品评价详情

商品回复列表

商品审核及操作记录

相关表结构

商品审核记录表

用于记录商品审核记录

```sql
create table pms_product_vertify_record
(
 id bigint not null auto_increment,
 product_id bigint comment '商品id',
 create_time datetime comment '创建时间',
 vertify_man varchar(64) comment '审核人',
 status int(1) comment '审核后的状态：0->未通过；2->已通过',
 detail varchar(255) comment '反馈详情',
 primary key (id)
);
```

商品操作记录表

用于记录商品操作记录

```sql
create table pms_product_operate_log
(
 id bigint not null auto_increment,
 product_id bigint comment '商品id',
 price_old decimal(10,2) comment '改变前价格',
 price_new decimal(10,2) comment '改变后价格',
 sale_price_old decimal(10,2) comment '改变前优惠价',
 sale_price_new decimal(10,2) comment '改变后优惠价',
 gift_point_old int comment '改变前积分',
 gift_point_new int comment '改变后积分',
 use_point_limit_old int comment '改变前积分使用限制',
 use_point_limit_new int comment '改变后积分使用限制',
 operate_man varchar(64) comment '操作人',
 create_time datetime comment '创建时间',
 primary key (id)
);
```

登录功能

前台用户目前涉及的表：

- ums_member
- ums_member_level
- ums_member_login_log
- ums_member_receive_address

ums_member 用户基础信息表

```
-- ----------------------------
-- Table structure for ums_member
-- ----------------------------
DROP TABLE IF EXISTS `ums_member`;
CREATE TABLE `ums_member` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `member_level_id` bigint(20) DEFAULT NULL,
  `username` varchar(64) DEFAULT NULL COMMENT '用户名',
  `password` varchar(64) DEFAULT NULL COMMENT '密码',
  `nickname` varchar(64) DEFAULT NULL COMMENT '昵称',
  `phone` varchar(64) DEFAULT NULL COMMENT '手机号码',
  `status` int(1) DEFAULT NULL COMMENT '帐号启用状态:0->禁用；1->启用',
  `create_time` datetime DEFAULT NULL COMMENT '注册时间',
  `icon` varchar(500) DEFAULT NULL COMMENT '头像',
  `gender` int(1) DEFAULT NULL COMMENT '性别：0->未知；1->男；2->女',
  `birthday` date DEFAULT NULL COMMENT '生日',
  `city` varchar(64) DEFAULT NULL COMMENT '所做城市',
  `job` varchar(100) DEFAULT NULL COMMENT '职业',
  `personalized_signature` varchar(200) DEFAULT NULL COMMENT '个性签名',
  `source_type` int(1) DEFAULT NULL COMMENT '用户来源',
  `integration` int(11) DEFAULT NULL COMMENT '积分',
  `growth` int(11) DEFAULT NULL COMMENT '成长值',
  `luckey_count` int(11) DEFAULT NULL COMMENT '剩余抽奖次数',
  `history_integration` int(11) DEFAULT NULL COMMENT '历史积分数量',
  PRIMARY KEY (`id`),
  UNIQUE KEY `idx_username` (`username`),
  UNIQUE KEY `idx_phone` (`phone`)
) ENGINE=InnoDB AUTO_INCREMENT=10 DEFAULT CHARSET=utf8 COMMENT='会员表';
```

ums_member_login_log 用户登录记录

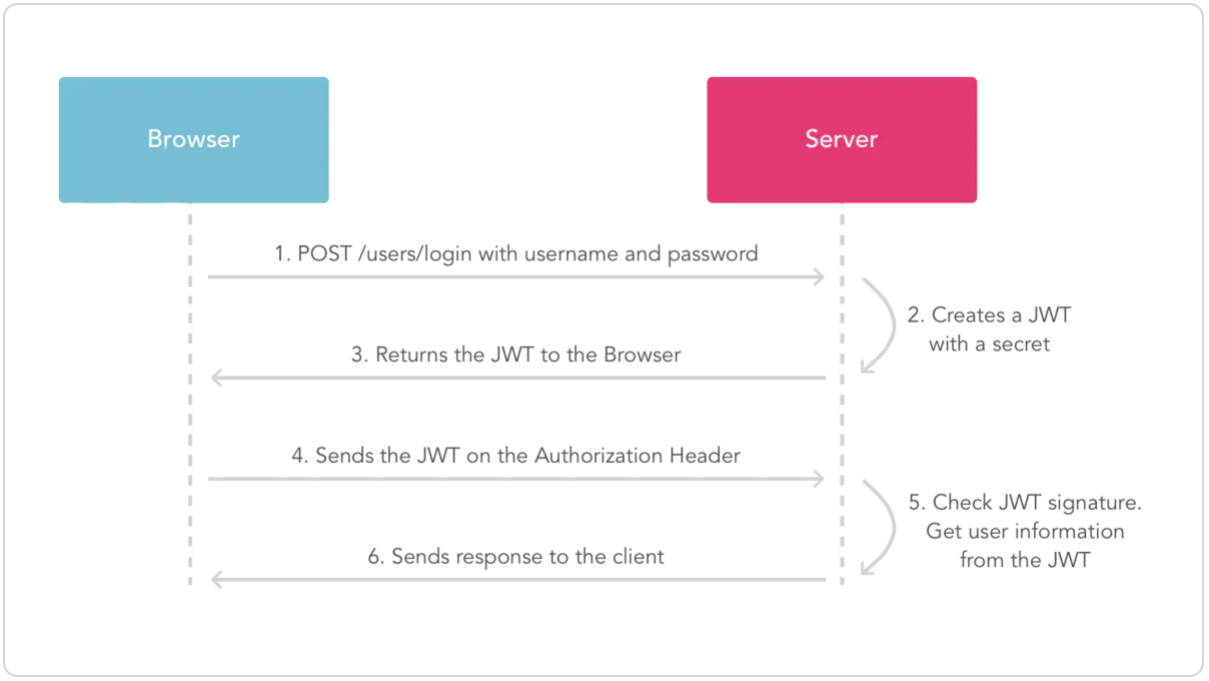
```
-- ----------------------------
-- Table structure for ums_member_login_log
-- ----------------------------
DROP TABLE IF EXISTS `ums_member_login_log`;
CREATE TABLE `ums_member_login_log` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `member_id` bigint(20) DEFAULT NULL,
  `create_time` datetime DEFAULT NULL,
  `ip` varchar(64) DEFAULT NULL,
  `city` varchar(64) DEFAULT NULL,
  `login_type` int(1) DEFAULT NULL COMMENT '登录类型：0->PC；1->android;2->ios;3->小程序',
  `province` varchar(64) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='会员登录记录';
```

ums_member_receive_address 用户收货地址表

```
-- ----------------------------
-- Table structure for ums_member_receive_address
-- ----------------------------
```

```sql
DROP TABLE IF EXISTS `ums_member_receive_address`;
CREATE TABLE `ums_member_receive_address` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `member_id` bigint(20) DEFAULT NULL,
  `name` varchar(100) DEFAULT NULL COMMENT '收货人名称',
  `phone_number` varchar(64) DEFAULT NULL,
  `default_status` int(1) DEFAULT NULL COMMENT '是否为默认',
  `post_code` varchar(100) DEFAULT NULL COMMENT '邮政编码',
  `province` varchar(100) DEFAULT NULL COMMENT '省份/直辖市',
  `city` varchar(100) DEFAULT NULL COMMENT '城市',
  `region` varchar(100) DEFAULT NULL COMMENT '区',
  `detail_address` varchar(128) DEFAULT NULL COMMENT '详细地址(街道)',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8 COMMENT='会员收货地址表';
```

jwt登录



HTTP/1.1
Header:
Authorization: Bearer **eyJhbGciOiJIUzI1NiIsI**.**eyJpc3MiOiJodHRwczotcGxlL**.**mFrs3Zo8eaSNcxiNfvRh9dqKP4F1cB**

1.**header**：对token类型和加密类型 进行base64加密得到；
{
"alg" :"HS256",
"typ" : "JWT"
}
编码base46：eyJhbGciOiJIUzI1NiIsI
2.**payload**：对一些有效信息 进行base64加密得到；
{
user_name:xxx
created:xxx
}
编码后base46：eyJpc3MiOiJodHRwczotcGxlL
前端解码： JSON.parse(window.atob("eyJpc3MiOiJodHRwczotcGxlL"))

3.**signature**：对base64加密后的header和base64加密后的payload使用.连接组成的字符串，再通过header中声明的加密方式进行加secret组合加密

正是它的存在，保证了整个jwt不被篡改

mFrs3Zo8eaSNcxiNfvRh9dqKP4F1cB

过程：

1. 用户使用账号和面发出post请求；

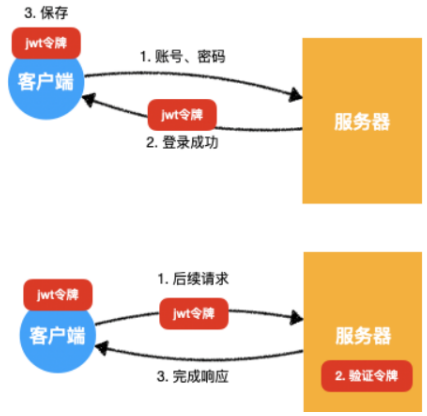
2. 服务器使用私钥创建一个jwt；

3. 服务器返回这个jwt给浏览器；

4. 浏览器将该jwt串在请求头中像服务器发送请求；

5. 服务器验证该jwt；

6. 返回响应的资源给浏览器。



实现：

1.添加jwt依赖：父maven中

```
<dependency>
 <groupId>io.jsonwebtoken</groupId>
 <artifactId>jjwt</artifactId>
 <version>${jwt.version}</version>
</dependency>
```

2.添加jwt加密解密 工具类

```
package com.tulingxueyuan.mall.common.util;

import cn.hutool.core.date.DateUtil;
import cn.hutool.core.util.StrUtil;
import io.jsonwebtoken.Claims;
import io.jsonwebtoken.Jwts;
import io.jsonwebtoken.SignatureAlgorithm;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Value;

import java.util.Date;
import java.util.HashMap;
import java.util.Map;

/**
 * JwtToken生成的工具类
 * JWT token的格式：header.payload.signature
 * header的格式（算法、token的类型）：
 * {"alg": "HS512","typ": "JWT"}
 * payload的格式（用户名、创建时间、生成时间）：
```

```
 * {"sub":"wang","created":1489079981393,"exp":1489684781}
 * signature的生成算法:
 * HMACSHA512(base64UrlEncode(header) + "." +base64UrlEncode(payload),secret)
 * Created on 2018/4/26.
 */
public class JwtTokenUtil {
 private static final Logger LOGGER = LoggerFactory.getLogger(JwtTokenUtil.class);
 private static final String CLAIM_KEY_USERNAME = "user_name";
 private static final String CLAIM_KEY_CREATED = "created";
 @Value("${jwt.secret}")
 private String secret;
 @Value("${jwt.expiration}")
 private Long expiration;
 @Value("${jwt.tokenHead}")
 private String tokenHead;

 /**
 * 根据负责生成JWT的token
 */
 private String generateToken(Map<String, Object> claims) {
 return Jwts.builder()
 .setClaims(claims)
 .setExpiration(generateExpirationDate())
 .signWith(SignatureAlgorithm.HS512, secret)
 .compact();
 }

 /**
 * 从token中获取JWT中的负载
 */
 private Claims getClaimsFromToken(String token) {
 Claims claims = null;
 try {
 claims = Jwts.parser()
 .setSigningKey(secret)
 .parseClaimsJws(token)
 .getBody();
 } catch (Exception e) {
 LOGGER.info("JWT格式验证失败:{}", token);
 }
 return claims;
 }

 /**
 * 生成token的过期时间
 */
 private Date generateExpirationDate() {
 return new Date(System.currentTimeMillis() + expiration * 1000);
 }

 /**
 * 从token中获取登录用户名
 */
 public String getUserNameFromToken(String token) {
 String username;
 try {
 Claims claims = getClaimsFromToken(token);
 username = claims.get(CLAIM_KEY_USERNAME,String.class);
 } catch (Exception e) {
 username = null;
```

```
}
return username;
}


/**
* 判断token是否已经失效
*/
private boolean isTokenExpired(String token) {
Date expiredDate = getExpiredDateFromToken(token);
return expiredDate.before(new Date());
}

/**
* 从token中获取过期时间
*/
private Date getExpiredDateFromToken(String token) {
Claims claims = getClaimsFromToken(token);
 return claims.getExpiration();
 }

 /**
 * 根据用户名生成token
 */
 public String generateUserNameStr(String username) {
 Map<String, Object> claims = new HashMap<>();
 claims.put(CLAIM_KEY_USERNAME,username);
 claims.put(CLAIM_KEY_CREATED, new Date());
 return generateToken(claims);
 }

 /**
 * 当原来的token没过期时是可以刷新的
 *
 * @param oldToken 带tokenHead的token
 */
 public String refreshHeadToken(String oldToken) {
 if(StrUtil.isEmpty(oldToken)){
 return null;
 }
 String token = oldToken.substring(tokenHead.length());
 if(StrUtil.isEmpty(token)){
 return null;
 }
 //token校验不通过
 Claims claims = getClaimsFromToken(token);
 if(claims==null){
 return null;
 }
 //如果token已经过期，不支持刷新
 if(isTokenExpired(token)){
 return null;
 }
 //如果token在30分钟之内刚刷新过，返回原token
 if(tokenRefreshJustBefore(token,30*60)){
 return token;
 }else{
 claims.put(CLAIM_KEY_CREATED, new Date());
 return generateToken(claims);
 }
 }
```

```
143
144  /**
145  * 判断token在指定时间内是否刚刚刷新过
146  * @param token 原token
147  * @param time 指定时间（秒）
148  */
149  private boolean tokenRefreshJustBefore(String token, int time) {
150  Claims claims = getClaimsFromToken(token);
151  Date created = claims.get(CLAIM_KEY_CREATED, Date.class);
152  Date refreshDate = new Date();
153  //刷新时间在创建时间的指定时间内
154  if(refreshDate.after(created)&&refreshDate.before(DateUtil.offsetSecond(created,time))){
155  return true;
156  }
157  return false;
158  }
159 }
```

3.在全局配置文件中添加jwt相关的配置

```
1 jwt:
2  secret: tuling-mall-portal #服务端私钥（一定不能泄露）
3  expiration: 86400 #过期时间 60*60*24 =一天
4  tokenHead: Bearer #jwt规范 #告诉客户端jwt令牌开头需要加的一个字符串
5  tokenHeader: Authorization #告诉客户端你要在请求头里面传什么参数名字
```

4.讲jwtUtils配置为bean

```
1 /**
2  * jwt工具类
3  * @return
4  */
5 @Bean
6 public JwtTokenUtil jwtTokenUtil(){
7  return new JwtTokenUtil();
8 }
```

5. 客户端发起登录请求（已完成）

6.服务端验证成功后加密jwt

```
1
2 @Value("${jwt.tokenHead}")
3 private String tokenHead;
4 @Value("${jwt.tokenHeader}")
5 private String tokenHeader;
6
7
8 @ApiOperation(value = "登录")
9 @RequestMapping(value = "/login", method = RequestMethod.POST)
10 @ResponseBody
11 public CommonResult login(@Validated UmsMember umsMemberParam) {
12  UmsMember login = memberService.login(umsMemberParam.getUsername(), umsMemberParam.getPassword());
13  if (login == null) {
14  return CommonResult.validateFailed("用户名或密码错误");
15  }
16  session.setAttribute(ComConstants.FLAG_MEMEBER_USER,login);
17
18  String token = jwtTokenUtil.generateUserNameStr(login.getUsername());
19
20  Map<String, String> tokenMap = new HashMap<>();
21  tokenMap.put("token",token);
22  tokenMap.put("tokenHead",token);
23  tokenMap.put("tokenHeader",token);
24  // jwt
25  return CommonResult.success(tokenMap);
26 }
```

7. 客户端接收拼接tokenHead+ jwt 存入cookie
8.后续客户端请求服务器 将cookie中jwt放入请求头中的Authorization
9.服务器拦截到请求头中的Authorization 解密jwt .从而拿到username 查询 是否存在用户
10.正常响应

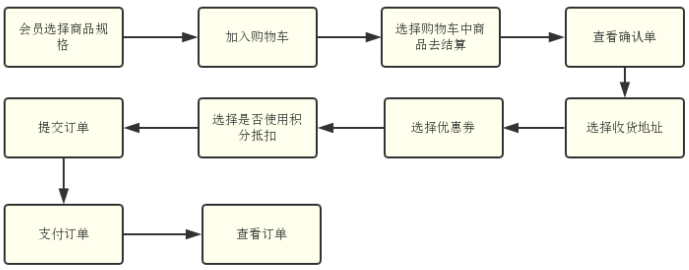
购物车

用于存储购物车中每个商品信息，可用于计算商品优惠金额。

```
create table oms_cart_item
(
 id bigint not null auto_increment,
 product_id bigint comment '商品的id',
 product_sku_id bigint comment '商品sku的id',
 member_id bigint comment '会员id',
 quantity int comment '购买数量',
 price decimal(10,2) comment '添加到购物车的价格',
 sp1 varchar(200) comment '销售属性1',
 sp2 varchar(200) comment '销售属性2',
 sp3 varchar(200) comment '销售属性3',
 product_pic varchar(1000) comment '商品主图',
 product_name varchar(500) comment '商品名称',
 product_brand varchar(200) comment '商品品牌',
 product_sn varchar(200) comment '商品的条码',
 product_sub_title varchar(500) comment '商品副标题（卖点）',
 product_sku_code varchar(200) comment '商品sku条码',
 member_nickname varchar(500) comment '会员昵称',
 create_date datetime comment '创建时间',
 modify_date datetime comment '修改时间',
 delete_status int(1) default 0 comment '是否删除',
 product_category_id bigint comment '商品的分类',
 product_attr varchar(500) comment '商品销售属性:[{"key":"颜色","value":"银色"},{"key":"容量","value":"4G"}]',
 primary key (id)
);
```

购物下单流程

整体流程示意图



移动端流程展示

- 会员选择商品规格

- 选择购物车中商品去结算

- 查看确认单

地址检索百度api：

http://lbsyun.baidu.com/index.php

本地域名映射到外网：

https://www.ngrok.cc/

ElementUI--auto

- 支付订单


支付订单
订单提交成功
请在00小时59分59秒内完成支付
支付金额 ￥180.00
支付方式
选择支付方式
支付宝支付
微信支付
银联支付
Apple Pay支付

- 支付成功

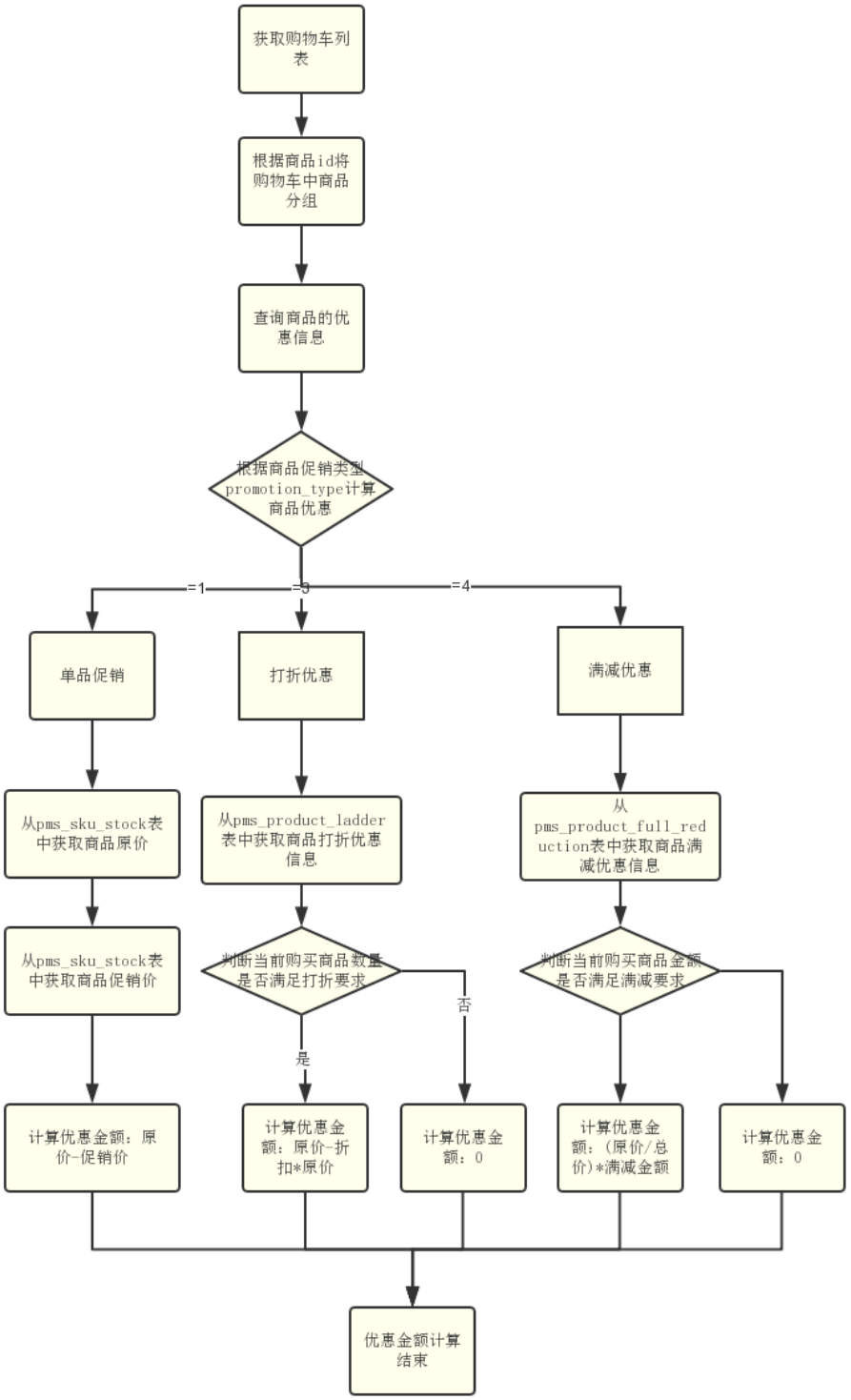
- 查看订单

[实现逻辑]

[加入购物车]

购物车的主要功能就是存储用户选择的商品信息及计算购物车中商品的优惠。

[购物车优惠计算流程]



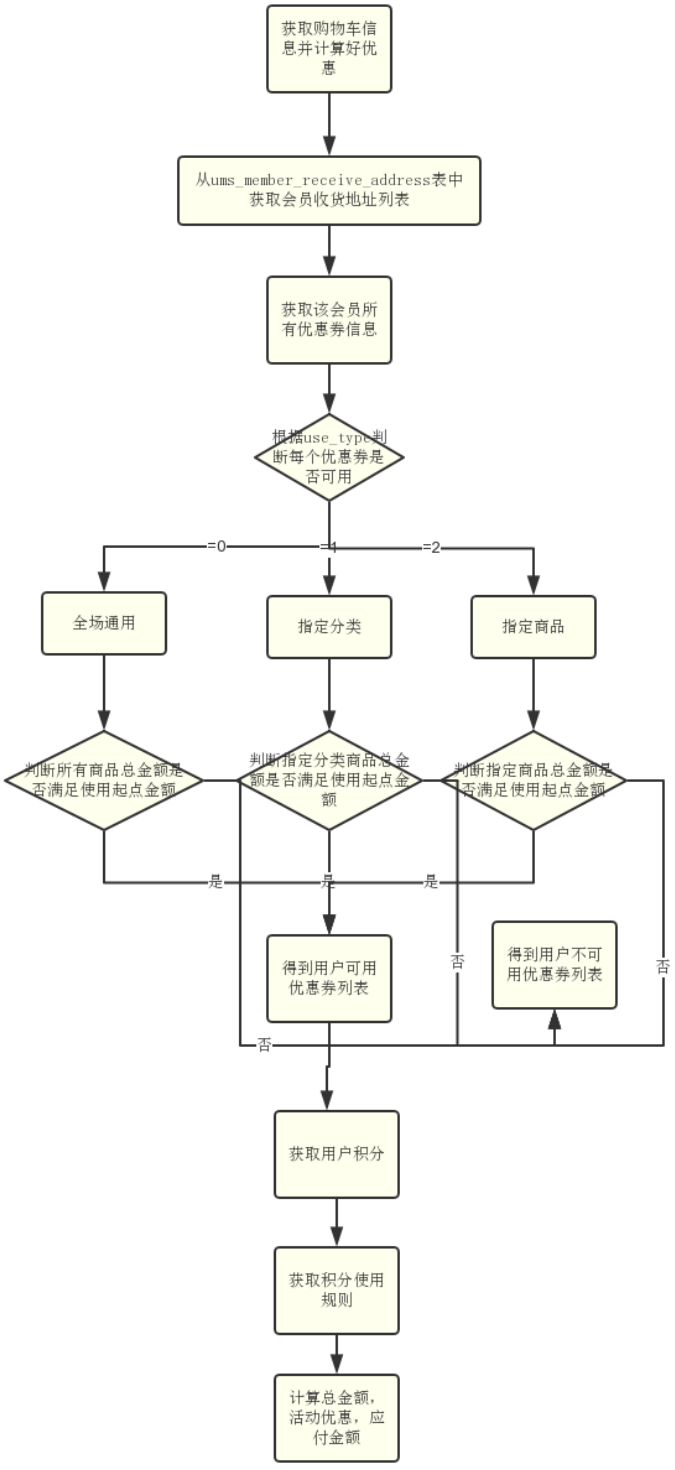
相关注意点

- 由于商品优惠都是以商品为单位来设计的，并不是以sku为单位设计的，所以必须以商品为单位来计算商品优惠；
- 代码实现逻辑可以参考OmsPromotionServiceImpl类中的calcCartPromotion方法。

生成确认单

确认单主要用于用户确认下单的商品信息、优惠信息、价格信息，以及选择收货地址、选择优惠券和使用积分。

生成确认单流程



相关注意点

- 总金额的计算：购物车中所有商品的总价；
- 活动优惠的计算：购物车中所有商品的优惠金额累加；
- 应付金额的计算：应付金额=总金额-活动优惠；
- 代码实现逻辑可以参考OmsPortalOrderServiceImpl类中的generateConfirmOrder方法。

生成订单

对购物车中信息进行处理，综合下单用户的信息来生成订单。

下单流程

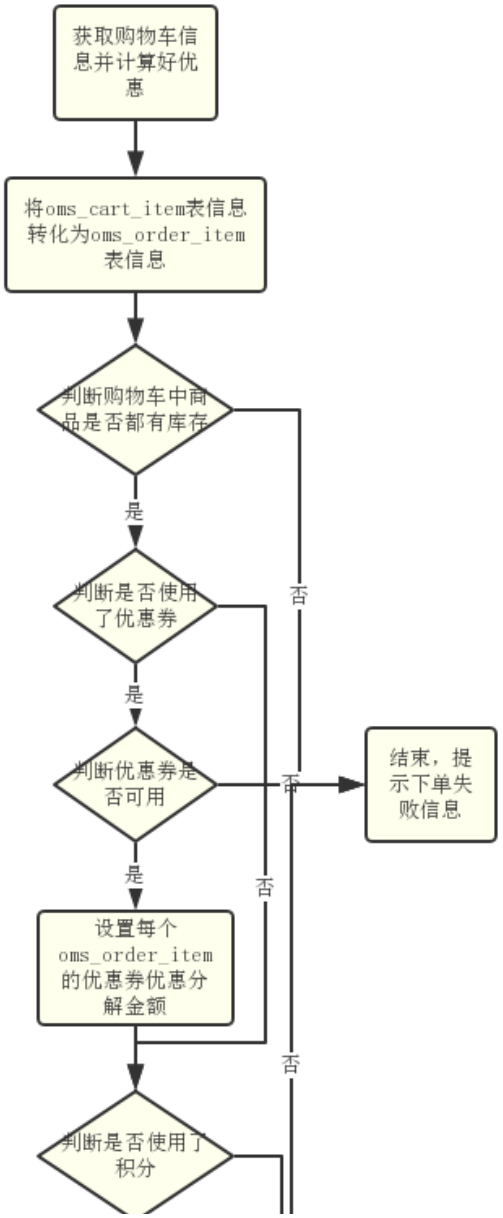
订单表

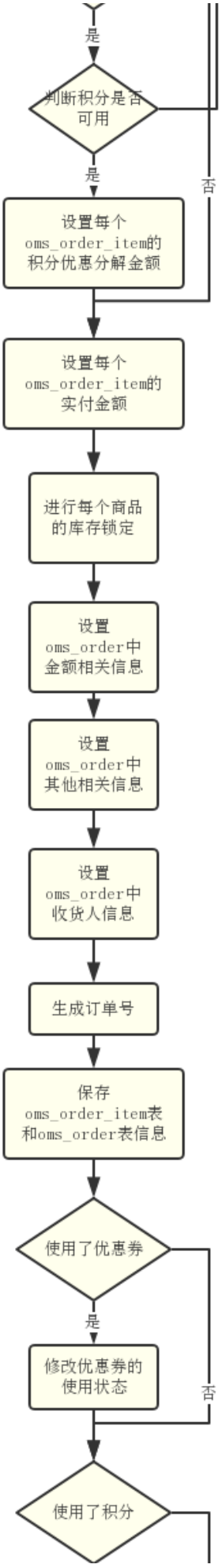
```
CREATE TABLE `oms_order` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT COMMENT '订单id',
  `member_id` bigint(20) NOT NULL,
  `coupon_id` bigint(20) DEFAULT NULL COMMENT '优惠券id',
  `order_sn` varchar(64) DEFAULT NULL COMMENT '订单编号',
  `create_time` datetime DEFAULT NULL COMMENT '提交时间',
  `member_username` varchar(64) DEFAULT NULL COMMENT '用户帐号',
  `total_amount` decimal(10,2) DEFAULT NULL COMMENT '订单总金额',
  `pay_amount` decimal(10,2) DEFAULT NULL COMMENT '应付金额（实际支付金额）',
  `freight_amount` decimal(10,2) DEFAULT NULL COMMENT '运费金额',
  `promotion_amount` decimal(10,2) DEFAULT NULL COMMENT '促销优化金额（促销价、满减、阶梯价）',
  `integration_amount` decimal(10,2) DEFAULT NULL COMMENT '积分抵扣金额',
  `coupon_amount` decimal(10,2) DEFAULT NULL COMMENT '优惠券抵扣金额',
  `discount_amount` decimal(10,2) DEFAULT NULL COMMENT '管理员后台调整订单使用的折扣金额',
  `pay_type` int(1) DEFAULT NULL COMMENT '支付方式：0->未支付；1->支付宝；2->微信',
  `source_type` int(1) DEFAULT NULL COMMENT '订单来源：0->PC订单；1->app订单',
  `status` int(1) DEFAULT NULL COMMENT '订单状态：0->待付款；1->待发货；2->已发货；3->已完成；4->已关闭；5->无效订单',
  `order_type` int(1) DEFAULT NULL COMMENT '订单类型：0->正常订单；1->秒杀订单',
  `delivery_company` varchar(64) DEFAULT NULL COMMENT '物流公司(配送方式)',
  `delivery_sn` varchar(64) DEFAULT NULL COMMENT '物流单号',
  `auto_confirm_day` int(11) DEFAULT NULL COMMENT '自动确认时间（天）',
  `integration` int(11) DEFAULT NULL COMMENT '可以获得的积分',
  `growth` int(11) DEFAULT NULL COMMENT '可以活动的成长值',
  `promotion_info` varchar(100) DEFAULT NULL COMMENT '活动信息',
  `bill_type` int(1) DEFAULT NULL COMMENT '发票类型：0->不开发票；1->电子发票；2->纸质发票',
  `bill_header` varchar(200) DEFAULT NULL COMMENT '发票抬头',
  `bill_content` varchar(200) DEFAULT NULL COMMENT '发票内容',
  `bill_receiver_phone` varchar(32) DEFAULT NULL COMMENT '收票人电话',
  `bill_receiver_email` varchar(64) DEFAULT NULL COMMENT '收票人邮箱',
  `receiver_name` varchar(100) NOT NULL COMMENT '收货人姓名',
  `receiver_phone` varchar(32) NOT NULL COMMENT '收货人电话',
  `receiver_post_code` varchar(32) DEFAULT NULL COMMENT '收货人邮编',
  `receiver_province` varchar(32) DEFAULT NULL COMMENT '省份/直辖市',
  `receiver_city` varchar(32) DEFAULT NULL COMMENT '城市',
  `receiver_region` varchar(32) DEFAULT NULL COMMENT '区',
  `receiver_detail_address` varchar(200) DEFAULT NULL COMMENT '详细地址',
  `note` varchar(500) DEFAULT NULL COMMENT '订单备注',
  `confirm_status` int(1) DEFAULT NULL COMMENT '确认收货状态：0->未确认；1->已确认',
  `delete_status` int(1) NOT NULL DEFAULT '0' COMMENT '删除状态：0->未删除；1->已删除',
  `use_integration` int(11) DEFAULT NULL COMMENT '下单时使用的积分',
  `payment_time` datetime DEFAULT NULL COMMENT '支付时间',
  `delivery_time` datetime DEFAULT NULL COMMENT '发货时间',
  `receive_time` datetime DEFAULT NULL COMMENT '确认收货时间',
  `comment_time` datetime DEFAULT NULL COMMENT '评价时间',
  `modify_time` datetime DEFAULT NULL COMMENT '修改时间',
  PRIMARY KEY (`id`)
)
```

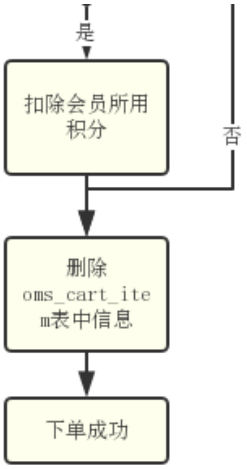
订单详情（购物车标）

```
CREATE TABLE `oms_order_item` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `order_id` bigint(20) DEFAULT NULL COMMENT '订单id',
  `order_sn` varchar(64) DEFAULT NULL COMMENT '订单编号',
  `product_id` bigint(20) DEFAULT NULL,
  `product_pic` varchar(500) DEFAULT NULL,
  `product_name` varchar(200) DEFAULT NULL,
  `product_brand` varchar(200) DEFAULT NULL,
  `product_sn` varchar(64) DEFAULT NULL,
```

```
  `product_price` decimal(10,2) DEFAULT NULL COMMENT '销售价格',
  `product_quantity` int(11) DEFAULT NULL COMMENT '购买数量',
  `product_sku_id` bigint(20) DEFAULT NULL COMMENT '商品sku编号',
  `product_sku_code` varchar(50) DEFAULT NULL COMMENT '商品sku条码',
  `product_category_id` bigint(20) DEFAULT NULL COMMENT '商品分类id',
  `sp1` varchar(100) DEFAULT NULL COMMENT '商品的销售属性',
  `sp2` varchar(100) DEFAULT NULL,
  `sp3` varchar(100) DEFAULT NULL,
  `promotion_name` varchar(200) DEFAULT NULL COMMENT '商品促销名称',
  `promotion_amount` decimal(10,2) DEFAULT NULL COMMENT '商品促销分解金额',
  `coupon_amount` decimal(10,2) DEFAULT NULL COMMENT '优惠券优惠分解金额',
  `integration_amount` decimal(10,2) DEFAULT NULL COMMENT '积分优惠分解金额',
  `real_amount` decimal(10,2) DEFAULT NULL COMMENT '该商品经过优惠后的分解金额',
  `gift_integration` int(11) DEFAULT '0',
  `gift_growth` int(11) DEFAULT '0',
  `product_attr` varchar(500) DEFAULT NULL COMMENT '商品销售属性:[{"key":"颜色","value":"颜色"},{"key":"容量","valu
e":"4G"}]',
  PRIMARY KEY (`id`)
)
```




是
判断积分是否可用
是
否
设置每个oms_order_item的积分优惠分解金额
设置每个oms_order_item的实付金额
进行每个商品的库存锁定
设置oms_order中金额相关信息
设置oms_order中其他相关信息
设置oms_order中收货人信息
生成订单号
保存oms_order_item表和oms_order表信息
使用了优惠券
是
否
修改优惠券的使用状态
使用了积分



相关注意点

1. 判断库存（如果没有库存直接提示）
2.保存订单主表oms_order信息 订单号
3.保存订单详情表oms_order_item(购物车转移)
4.锁定库存（如果用户超过10分钟没有支付-恢复库存）
5.删除对应购物车

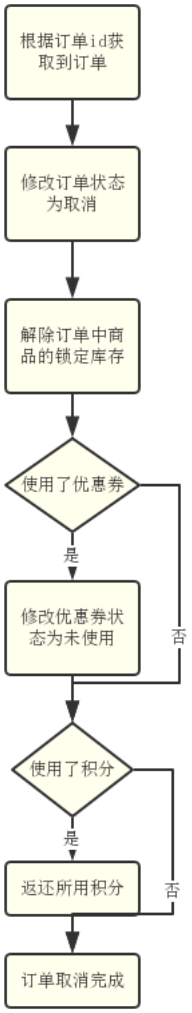
- 库存的锁定：库存从获取购物车优惠信息时就已经从`pms_sku_stock`表中查询出来了，lock_stock字段表示锁定库存的数量，会员看到的商品数量为真实库存减去锁定库存；
- 优惠券分解金额的处理：对全场通用、指定分类、指定商品的优惠券分别进行分解金额的计算：
  - 全场通用：购物车中所有下单商品进行均摊；
  - 指定分类：购物车中对应分类的商品进行均摊；
  - 指定商品：购物车中包含的指定商品进行均摊。
- 订单中每个商品的实际支付金额计算：原价-促销优惠-优惠券抵扣-积分抵扣，促销优惠就是购物车计算优惠流程中计算出来的优惠金额；
- 订单号的生成：使用redis来生成，生成规则:8位日期+2位平台号码+6位以上自增id；
- 优惠券使用完成后需要修改优惠券的使用状态；
- 代码实现逻辑可以参考OmsPortalOrderServiceImpl类中的generateOrder方法。

取消订单

订单生成之后还需开启一个延时任务来取消超时的订单。

订单取消流程

1. 获取规定时间假设：10分钟内未支付的所有的订单， 系统自动去做的： 任务调度（定时时间） 1分钟
   用户下单： 12:00:01
   系统调度：1分钟一次 12:11 问题：时间精度 ， 明确：redis\ mq
2.状态修改取消
3.解除锁定库存



任务调度方式：

1. Spring task ：

只需要

1.引入spring-boot-start-web

2.启用注解`@EnableScheduling`

3.第三步添加定时器：

```
@Scheduled(cron="0 0/1 * ? * ?")
```

2. quartz 支持分布式下的任务调度

3.xxl_job 支持分布式下的任务调度

@Scheduled注解固定调度时间设置

- **Cron表达式详解**

一、结构

cron从左到右（用空格隔开）：秒 分 小时 月份中的日期 月份 星期中的日期 年份

年份可省略

二、各字段的含义

字段 取值范围 允许的特殊字符

秒		0-59		, - * /
分		0-59		, - * /
小时		0-23		, - * /
日期		1-31		, - * ? / L W C
月份		1-12 或者 JAN-DEC		, - * /
星期		1-7 或者 SUN-SAT		, - * ? / L C #
年（可选）		留空，1970-2099		, - * /

每一个域都使用数字，但还可以出现如下特殊字符，它们的含义是：

（1）*****：表示匹配该域的任意值。“*”在子表达式（月）里表示每个月的含义，“*”在子表达式（天（星期））表示星期的每一天。

（2）**?**：只能用在DayofMonth和DayofWeek两个域，表示不指定值。当2个子表达式其中之一被指定了值以后，为了避免冲突，需要将另一个子表达式的值设为“?”。

（3）**-**：表示范围。例如在Minutes域使用5-20，表示从5分到20分钟每分钟触发一次

（4）**/**：表示起始时间开始触发，然后每隔固定时间触发一次。例如在Minutes域使用5/20,则意味着5分钟开始触发一次，每隔20分钟触发一次，即25，45等分别触发一次.

（5）**,**：表示列出枚举值。例如：在Minutes域使用5,20，则意味着在5分钟和20分钟各触发一次。

（6）**L**：表示最后，只能出现在DayofWeek和DayofMonth域。在DayofWeek中，“L”表示一个星期的最后一天，也就是SAT；在DayofMonth中，“L”表示一个月的最后一天。如果在“L”前有具体的内容，它就具有其他的含义了。如果在DayofWeek域使用5L，意味着在最后的一个星期四触发；如果在DayofMonth域使用6L表示一个月的倒数第６天，“FRIL”表示一个月的最后一个星期五。

（7）**W**：表示有效工作日(周一到周五)，只能出现在DayofMonth域，系统将在离指定日期的最近的有效工作日触发事件。例如：在DayofMonth使用4W，如果4日是星期六，则将在最近的工作日：星期五，即3日触发；如果4日是星期天，则在5日(周一)触发；如果4日在星期一到星期五中的一天，则就在4日触发。另外一点，W的最近寻找不会跨过月份 。

（8）**LW**：这两个字符可以连用，表示在某个月最后一个工作日。

（9）**#**：表示每个月第几个星期几，只能出现在DayofMonth域。例如在5#2，表示每个月的第二个星期四。

使用方法

可以通过注解@Scheduled设置调度时间
eg. @Scheduled(cron = “0 10 0 1 * ?”)
表示每月1号0点10分触发调度任务

支付：

1.根据不同的支付平台请求对应的第三方支付接口
2.当面支付（生成二维码）
3.用户支付
4.支付完成
5.更新订单状态和支付方式
6.清除锁定库存，扣除实际库存
7.二维码

支付宝支付：

图灵学院 楼兰老师：https://www.bilibili.com/video/BV1ry4y117ZU?p=1

进入支付宝开放平台：https://open.alipay.com/platform/home.htm
进入能力中心-->支付能力--->电脑网站支付：

https://nengli.alipay.com/abilityprod/detail?abilityCode=SM0100000000000001013

申请为开发者：

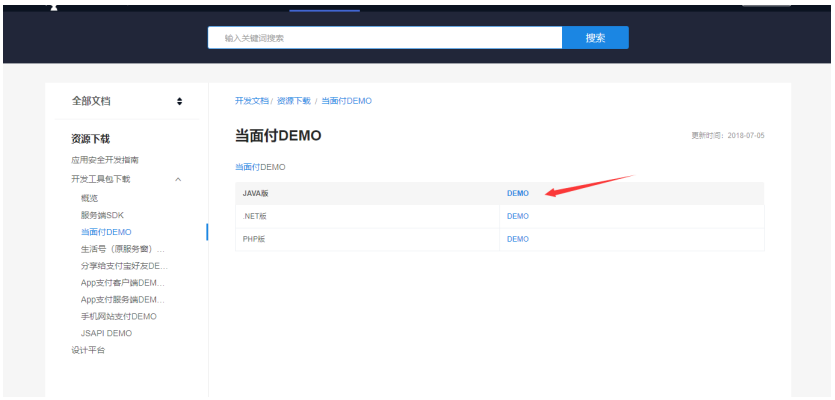
appid

getwaye

支付宝公钥，　商户公钥对应的私钥

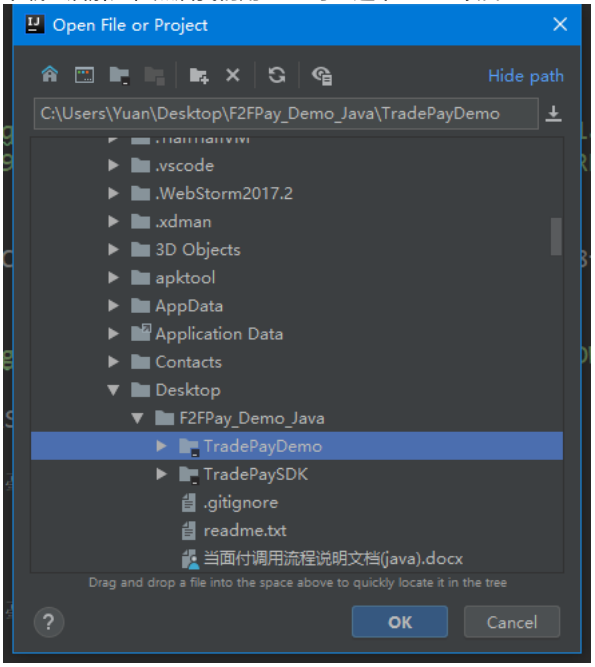
1、下载导入项目
https://docs.open.alipay.com/54/104506 打开支付宝接口官网：

它是MyEclipse项目，如果要在idea中打开：https://blog.csdn.net/lxfHaHaHa/article/details/78285102



我们下载Java版Demo

下载之后解压，然后我们用IDEA导入这个Demo项目~

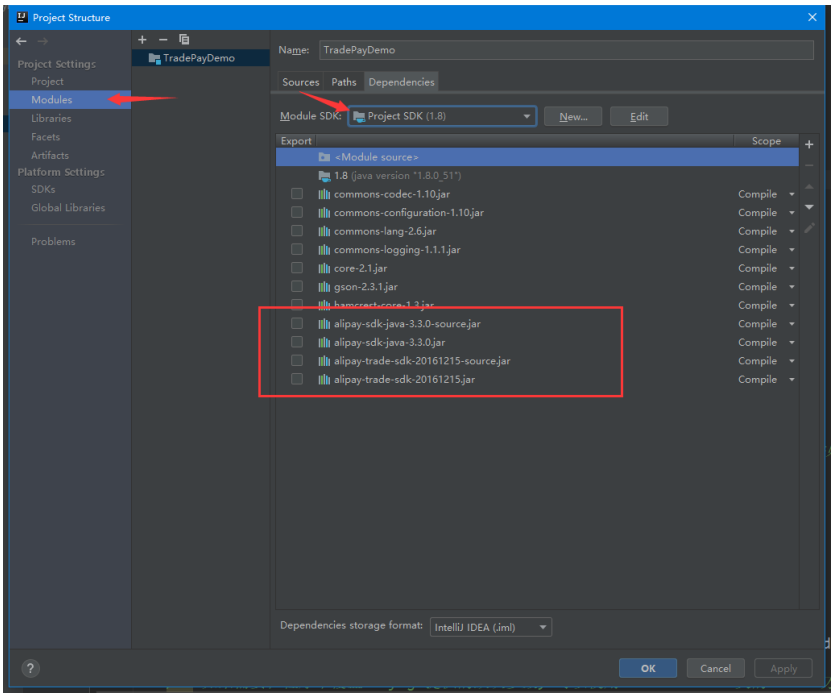


然后，我们下载一个我们后面需要生成生成RSA密钥的工具:https://docs.open.alipay.com/291/105971

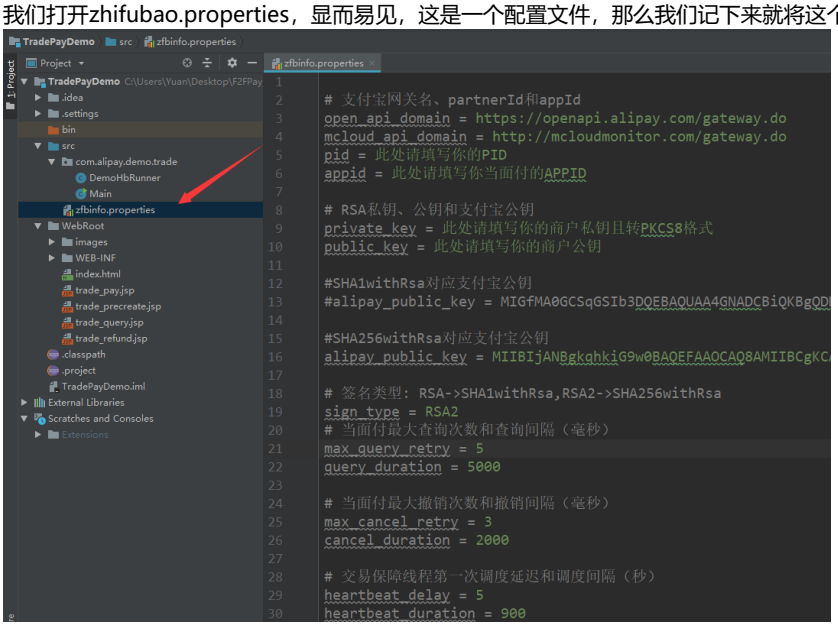
由于我是在Win平台开发，所以下载自己操作系统对应的版本就行：

下载之后我们就先放一边啦~

导入项目之后，我们要查看自己导入的项目是否报错，如果出现报错，可能要调整一下自己的项目环境：



好，配置好项目时候，我们先不着急运行项目，
我们打开zhifubao.properties，显而易见，这是一个配置文件，那么我们记下来就将这个配置文件的相关配置配好~



####2、配置好配置文件
由于我们测试的时候沙箱环境：
首先：

1、打开支付宝API官网的沙箱位置：https://openhome.alipay.com/platform/appDaily.htm?tab=info

打开这个网址，我们就会看到下面界面：

首先我们看到相关参数，我们不管，我们一步步按照官网给的Demo里面的配置文件一个个将配置文件配好即可~

2、打开zhifubao.properties配置文件，我们就会看到下面代码：

首先我们先配置最上面的四行：

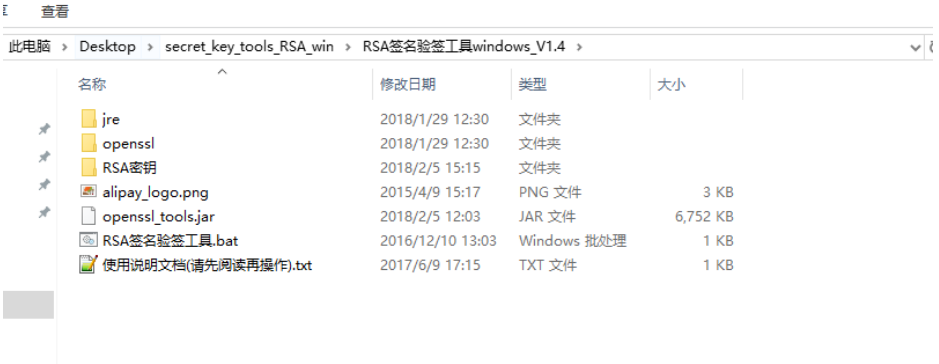
```
# 支付宝网关名、partnerId和appId
open_api_domain = https://openapi.alipay.com/gateway.do
mcloud_api_domain = http://mcloudmonitor.com/gateway.do
pid = 此处请填写你的PID
appid = 此处请填写你当面付的APPID
```

这四行我们根据沙箱环境里面给的对应参数来配置

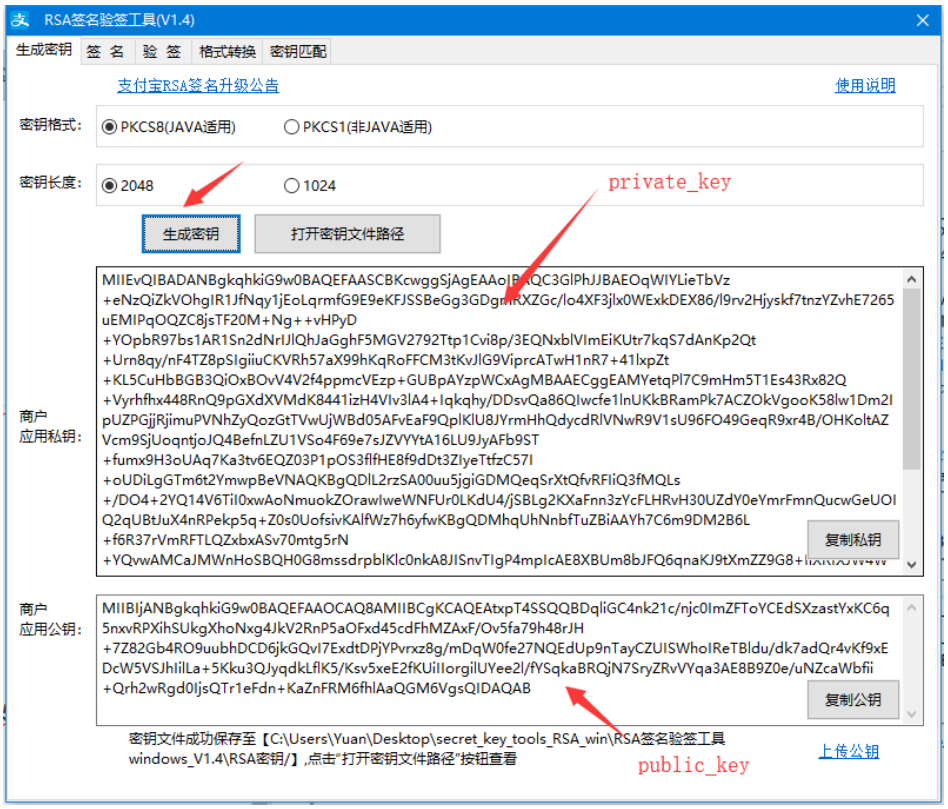
mcloud_api_domain这个参数我们不需要改变~

3、接下来就是配置公钥和私钥了

我们打开上面再这个链接下载的支付宝官网提供的公钥私钥生成工具：https://docs.open.alipay.com/291/105971
解压之后：



双击：RSA签名验签工具.bat
由于我们这次选择的是RSA2密钥方式：所以我们选择密钥长度为2048的方式：



点击生成密钥
接下来我们配置Demo中配置文件的下两行

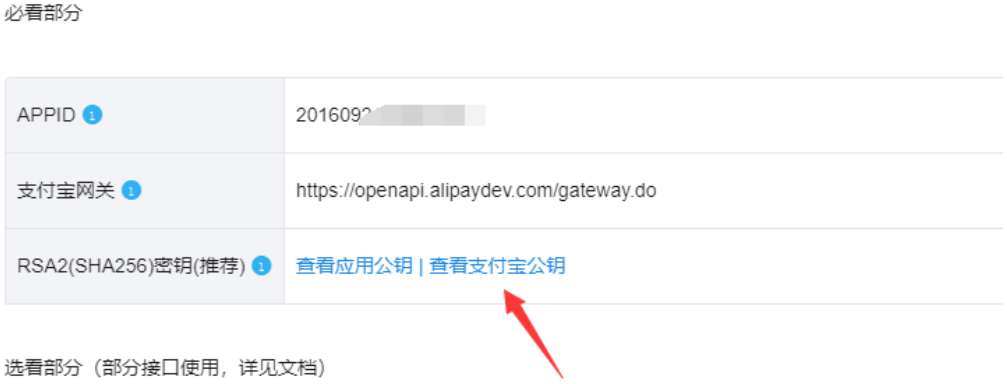
```
# RSA私钥、公钥和支付宝公钥
private_key = 此处请填写你的商户私钥且转PKCS8格式
public_key = 此处请填写你的商户公钥
```

将上面密钥生产工具生成的私钥和公钥复制到对应的地方即可：

4、下面一步，我们配置支付宝公钥，这一步我们要回到沙箱环境中：

对这一行进行操作，首先我们复制上面密钥工具生成的公钥，然后再支付宝沙箱环境页面，点击查看公钥，然后点击修改，删除原来的，然后将我们刚才在密钥生成工具生成的公钥粘贴到里面：

点击保存：
然后点击查看支付宝公钥：



然后复制里面的支付宝公钥，下一步回到我们的Demo项目中，打开我们的zhifubao.properties配置文件，将复制的支付宝公钥放到下面参数的配置上，记得把原来配置文件里默认的删除掉，

```
#SHA256withRsa对应支付宝公钥
alipay_public_key =
```

这个是默认注释的，因为我们选中的是#SHA256withRsa秘钥方式，
下面的参数选择默认的就差不多啦。。
到此，我们的支付宝Demo的配置文件算是配置好了~

####3、运行Demo:
配置好配置文件之后，我们运行一下Demo的Main函数：发现项目可以正常跑起来了（如果你发发现你的项目报错，可能就是配置配置文件相关地方配置错了，根据错误提示一步步排查即可）

支付宝的maven依赖:

```
<!-- 支付宝支付 -->
<dependency>
  <groupId>com.alipay.sdk</groupId>
  <artifactId>alipay-sdk-java</artifactId>
  <version>3.3.87.ALL</version>
  <exclusions>
  <exclusion>
  <artifactId>commons-logging</artifactId>
```

```
 <groupId>commons-logging</groupId>
  </exclusion>
  </exclusions>
</dependency>


<dependency>
  <groupId>commons-lang</groupId>
  <artifactId>commons-lang</artifactId>
  <version>2.6</version>
</dependency>
<dependency>
  <groupId>commons-configuration</groupId>
  <artifactId>commons-configuration</artifactId>
  <version>1.10</version>
  <exclusions>
  <exclusion>
  <artifactId>commons-logging</artifactId>
  <groupId>commons-logging</groupId>
  </exclusion>
  </exclusions>
</dependency>
<dependency>
  <groupId>commons-codec</groupId>
  <artifactId>commons-codec</artifactId>
  <version>1.11</version>
</dependency>
<dependency>
  <groupId>com.google.zxing</groupId>
  <artifactId>core</artifactId>
  <version>3.2.1</version>
</dependency>
<dependency>
  <groupId>org.hamcrest</groupId>
  <artifactId>hamcrest-core</artifactId>
  <version>1.3</version>
  <scope>test</scope>
</dependency>
<dependency>
  <groupId>com.google.code.gson</groupId>
  <artifactId>gson</artifactId>
  <version>2.8.5</version>
</dependency>
```