

尚品汇商城复习

版本：V 1.0

支付模块

一、支付宝介绍

1 支付宝简介

[支付宝（中国）网络技术有限公司](#)^[1] 是国内的第三方支付平台，致力于提供“简单、安全、快速”的支付解决方案^[2]。支付宝公司从 2004 年建立开始，始终以“信任”作为产品和服务的核心。旗下有“支付宝”与“支付宝钱包”两个独立品牌。自 2014 年第二季度开始成为当前全球最大的[移动支付](#)厂商。

当用户提交订单会跳转到选择支付渠道页面！



订单提交成功，请您及时付款，以便尽快为您发货~~

请您在提交订单4小时之内完成支付，超时订单会自动取消。订单号：145687

应付金额：¥17,654

重要说明：

1. 品优购商城支付平台目前支持[支付宝](#)支付方式。
2. 其它支付渠道正在调试中，敬请期待。
3. 为了保证您的购物支付流程顺利完成，请保存以下支付宝信息。

支付宝账户信息：（很重要，请保存！！）

- 支付帐号：duqthf1038@sandbox.com
- 密码：111111
- 支付密码：111111

支付平台



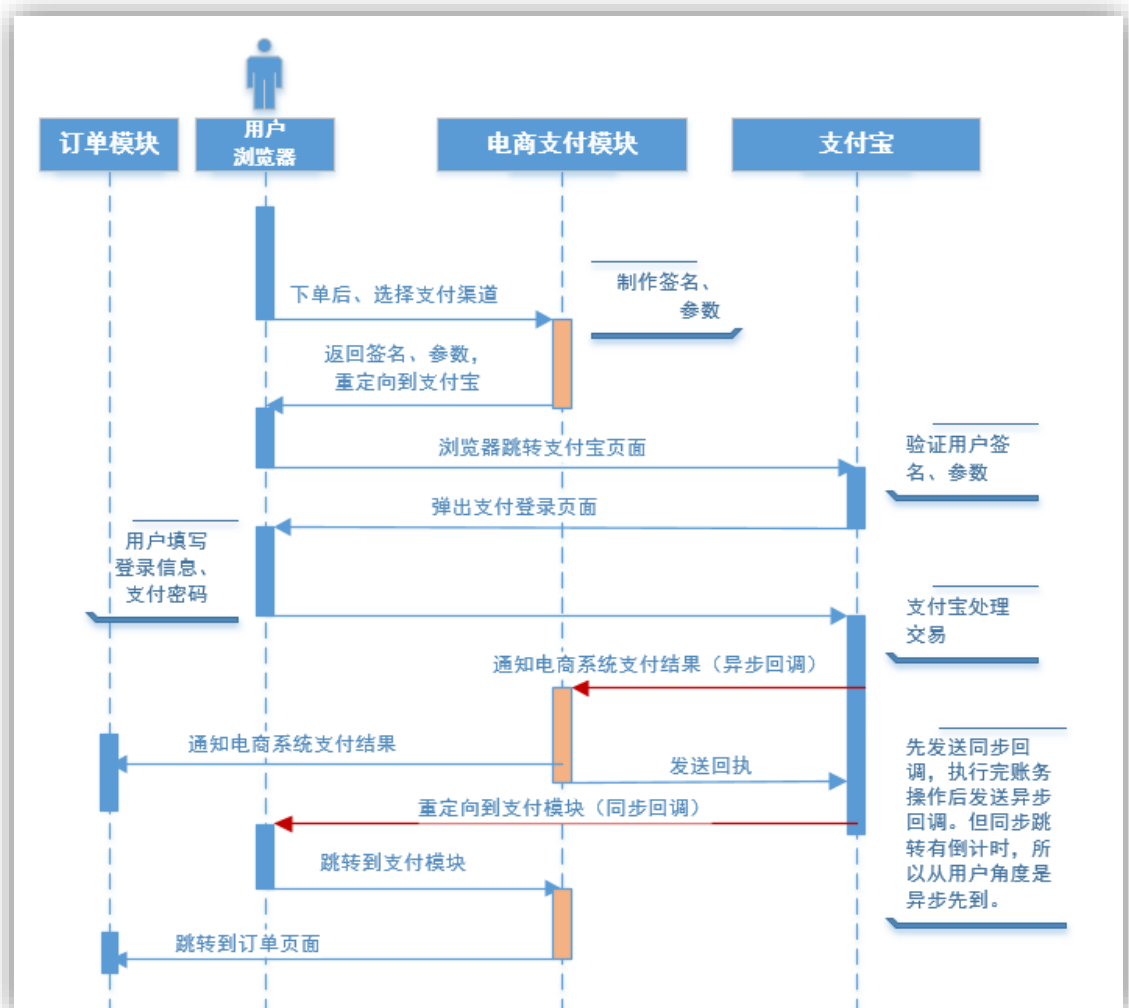
当用户点击立即支付时生成支付的二维码



使用支付宝 app 进行扫码支付



2 过程分析



3 对接支付宝的准备工作

1、申请条件

1. 企业或个体工商户可申请；
2. 提供真实有效的营业执照，且支付宝账户名称需与营业执照主体一致；
3. 网站能正常访问且页面信息有完整商品内容；
4. 网站必须通过 ICP 备案，个体户备案需与账户主体一致。
(团购类网站不支持个体工商户签约)

支付手续费

电脑网站支付

服务名称	费率	服务期限
单笔费率	0.6%	1年

费率说明：

助力中小商户，从签约日至2018.12.31日优惠费率为0.55%（不包括特殊行业）

特殊行业：休闲游戏；网络游戏点卡、游戏渠道代理；游戏系统商；网游周边服务、交易平台；网游运营商（含网页游戏）

4 申请步骤：

- 1、支付宝商家中心中申请 <https://www.alipay.com/>
- 2、<https://b.alipay.com/signing/productSetV2.htm?mrchportalwebServer=https%3A%2F%2Fmrchportalweb.alipay.com>



一个工作日后登录到蚂蚁金服开发者中心中：



可以查看到一个已经签约上线的应用。其中非常重要是这个 **APPID**，需要记录下来之后的程序中要用到这个参数。

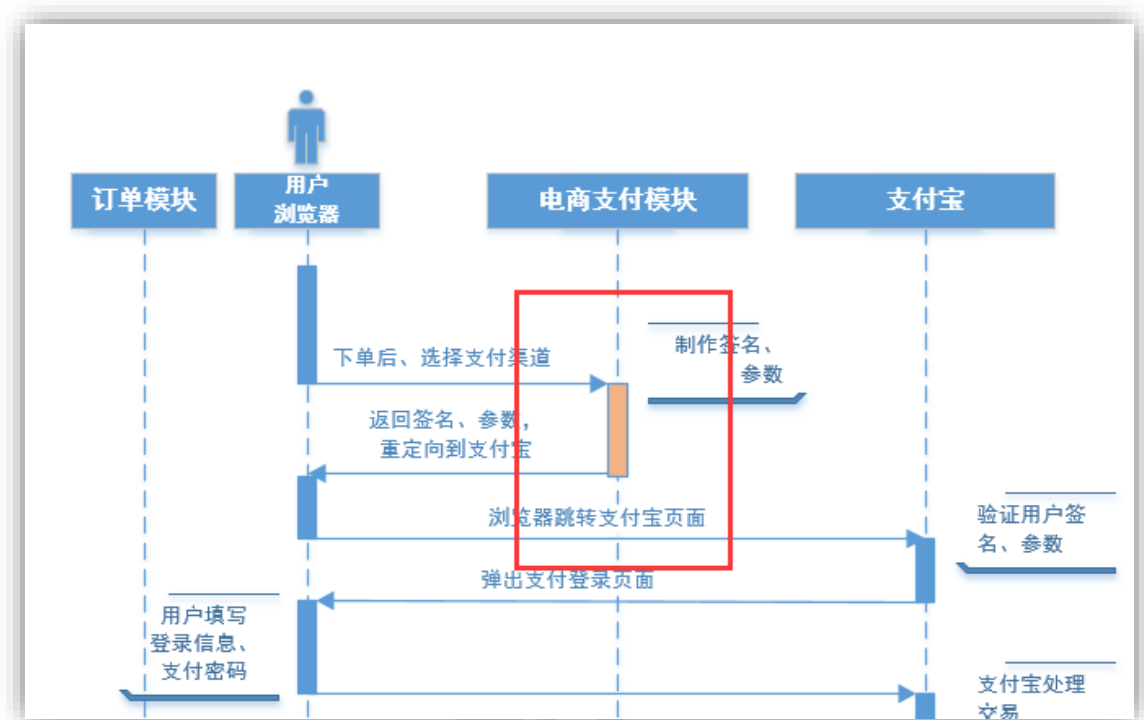
点击查看



到此为止，电商网站可以访问支付宝的最基本的准备已经完成。

支付宝开发手册：<https://docs.open.alipay.com/270/105900/>

二、支付功能实现



1 思路分析



1. 将支付数据保存到数据库，以便跟支付宝进行对账
2. 生成要支付的二维码

生成二维码需要的参数列表请参考官方文档

https://docs.open.alipay.com/api_1/alipay.trade.page.pay/

2 保存支付信息 - 制作实体类

表结构 payment_info

payment_info(支付信息表)	
 id	编号
 out_trade_no	对外业务编号
 order_id	订单编号
 payment_type	支付类型 (微信 支付宝)
 trade_no	交易编号
 total_amount	支付金额
 subject	交易内容
 payment_status	支付状态
 create_time	创建时间
 callback_time	回调时间
 callback_content	回调信息

id	主键自动生成
out_trade_no	订单中已生成的对外交易编号。订单中获取
order_id	订单编号
payment_type	支付类型 (微信与支付宝)
trade_no	交易号, 回调时生成
total_amount	订单金额。订单中获取
subject	交易内容。利用商品名称拼接。
payment_status	支付状态, 默认值未支付。
create_time	创建时间, 当前时间。
callback_time	回调时间, 初始为空, 支付宝异步回调时记录
callback_content	回调信息, 初始为空, 支付宝异步回调时记录

PaymentStatus

```

public enum PaymentStatus {

    UNPAID("未支付"),
    PAID("已支付"),
    PAY_FAIL("支付失败"),
    CLOSED("已关闭");

    private String name ;

    PaymentStatus(String name) {
        this.name=name;
    }
}

```

```
}  
}
```

3 编写支付宝支付接口

4.1 制作 AlipayClient 工具类

在 resource 中添加 alipay.properties

```
alipay_url=https://openapi.alipay.com/gateway.do
```

```
app_id=2018020102122556
```

```
app_private_key=MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBAcwggSjAgEAAoIBAQCdQek  
nhM2rhiGAH6V01jxn3rAWIdzduTEQuteTfwjnZtvMhQPuuN1b/88D5yMuaZhZNFeUdWb+Sm  
tP9DAzAWWgnT13T0YhJcxP6txm7JBRrjadCRt+LOFxPiPQk5t9fH7yXjw9i4uMDsNJeTncr  
VZ/AZYrk0ESC9anJR8XeuBc3HE8T4fq1KK135j1umIWrPbPNQhKGXaGcOnpiaX09qYYUSP/  
tnrjNYXH0so0yBs4YT1+LLX2TJ12p3n/oX6HnL4zQgtN5k4QasHP7CIig1ngcVQGfWsMm4d  
jI9KXNXvGLQPfMQEmyb71mM50Cd11MtAc60aIAymhSv2h0LNIuyodAgMBAAECggEAE05/P5  
mGm4Q1KI2n8u8K1neqovASe1kG/BNFjkYB+VBR80Ar4TfbepPvAyRuFap+5xN/yMz14VcBJ  
kRWtufVhEdHNxJV7w/wUIncIGhGEYFFMVbZWhTrbQH6TiUp6TC9dCmc6vD1CKPRkFj+YGB  
XT01Py3LzBa0TYNyCbszyhthrgkpuFYbB0R93IPvvBh5NJFXQytwNb2oVopC9AQWviqnZUZ  
cT0eJ087dQ1WLPa6b1BD8DP1PUq0Ldr6pgKf0bFxiJ8+87D1JznRfdEsbqZLS7jagdw5tLr  
71WJpctIGPqKpgvajfePP/lj3eY82BKQB+aTw0zmAiB05Yes4LgQKBgQDq3EiQR8J1MEN2r  
piLt1WvDYyVkvUgOY70d//fRPGAmbstbe4TzGBPr8E+z267bHAWLaWtHkfx6muFhN1x68oz  
EUWk/nZq0smWnuPdcy4E7Itbk36W2FF/r0ZB7j5dd1C9byrxDSNgc9/FA/CU+i5KVQpLYf  
sk2dvomvu0aFVQKBgQCraXpxzMmsBx4127LsZD05bxfxb6nqzyK4NPe0VaGiRg8oaCwczc  
Lz1J5iRqC9QeEwsSt4XU1sYBMTcsFpA0apZpm3prH2HJRx/isNENesaHcihF0mM0WxU3xy  
RvWSDeZV5A1Zy1ZEJ+p17DGwb2j+yo2uBrDNXBgBWEzXwiRqQKBgBdXfvsHtqKQz10QHgbe  
LGy+K1SrheMy9Sc9s7cLkqB/oWPNZfifugEceW71jGqh5y29EZb3yGoDyPwsxwi4Rxr2H3a  
7Nyd81T4bwkdyt+MTYvIR4WW6T7chhgyMsbP2GyYIUzsrdbWUnrCRXNOSJTGpksyY0sZHC+  
OGcMp/EQ4VAoGBAIISSVL/pm1+/UK7U1ukcced8JpKNLM0uVD1CJ50eHHOHgR4e0owrWYfi  
oxisejLjB1J6AwvL2g0w2T3qKKKVN2JOM4u1U5/w314+KwygqaWowizTogEQJPd5ta52ADT  
zjTzSD/t6nByd+YHAWLhc4lyt0bmj6pf68VBb8/upm75AoGAGAYz79IVHp9eppykufjNcWu  
6okkG8tZnzuyaWKW/CuKKBWMatk0vcyQ1fJfxIBccoQrBuYyXBdcpPuZ/ys2C25pNrKACuh  
IKNgnMc0floJoYEfJzetw/3cIimWu4NJzVQ0aojaGA58oo2+fub43Xn25Jq4rvSVe3oLdb5  
xWkw5Q=
```

```
alipay_public_key=MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAhkZi6W0wn  
/prX+NIIF9ATb5Z8ReKK4hFYtBrweDfGHD1mNW7YIZY4G5hE7S2Sry8eFX1FgS1BW1J4fVn  
DaK9MkVThpwE2H65ooV1K/wLuyPqovIVpMt/utva5Ayuzv7eQOWK45FdLDND1K8QLoBko6S  
S+YbnWnf7a+mr4NAS4UFC1pfe8Byqe8XIra02Cg4Ko5Y5schX39rOAH8G1LldgqQRYVQ2dC  
nkIQ+L+I4Cy9Mvw3rIkTwt3MBU+AqREXY4r5Bn6cmmX/9MAJbFqroFGiUAqG+qbjtCzAzgN  
PfuiD0zXgt/YYjMQMzcK75B0mwnY0am2ajODUSQn8Xybsa7wQIDAQAB
```



```
return_payment_url=http://api.gmall.com/api/payment/alipay/callback/return
notify_payment_url=http://xsiv7k.natappfree.cc/api/payment/alipay/callback/notify
return_order_url=http://payment.gmall.com/pay/success.html
```

创建配置类

```
package com.atguigu.gmall.payment.config;

@Configuration
@PropertySource("classpath:alipay.properties")
public class AlipayConfig {

    @Value("${alipay_url}")
    private String alipay_url;

    @Value("${app_private_key}")
    private String app_private_key;

    @Value("${app_id}")
    private String app_id;

    public final static String format="json";
    public final static String charset="utf-8";
    public final static String sign_type="RSA2";
    public static String return_payment_url;
    public static String notify_payment_url;
    public static String return_order_url;
    public static String alipay_public_key;

    @Value("${alipay_public_key}")
    public void setAlipay_public_key(String alipay_public_key) {
        AlipayConfig.alipay_public_key = alipay_public_key;
    }
    @Value("${return_payment_url}")
    public void setReturn_url(String return_payment_url) {
        AlipayConfig.return_payment_url = return_payment_url;
    }
    @Value("${notify_payment_url}")
    public void setNotify_url(String notify_payment_url) {
        AlipayConfig.notify_payment_url = notify_payment_url;
    }

    @Value("${return_order_url}")
    public void setReturn_order_url(String return_order_url) {
        AlipayConfig.return_order_url = return_order_url;
    }
}
```

```
}
@Bean
public AlipayClient alipayClient(){
    // AlipayClient alipayClient = new
    DefaultAlipayClient("https://openapi.alipay.com/gateway.do", APP_ID,
    APP_PRIVATE_KEY, FORMAT, CHARSET, ALIPAY_PUBLIC_KEY, SIGN_TYPE); //
    获得初始化的AlipayClient
    AlipayClient alipayClient=new
    DefaultAlipayClient(alipay_url,app_id,app_private_key,format,charset
    , alipay_public_key,sign_type );
    return alipayClient;
}
}
```

4.2 编写支付宝下单接口

实现类

```
package com.atguigu.gmall.payment.service.impl;
@Service
public class AlipayServiceImpl implements AlipayService {

    @Autowired
    private PaymentService paymentService;

    @Autowired
    private OrderFeignClient orderFeignClient;

    @Autowired
    private AlipayClient alipayClient;

    @Override
    public String createaliPay(Long orderId) throws
    AlipayApiException {
        OrderInfo orderInfo =
        orderFeignClient.getOrderInfo(orderId);
        // 保存交易记录
        paymentService.savePaymentInfo(orderInfo,
        PaymentType.ALIPAY.name());
        // 生产二维码
        AlipayTradePagePayRequest alipayRequest = new
        AlipayTradePagePayRequest();//创建API 对应的request
        // 同步回调
        alipayRequest.setReturnUrl(AlipayConfig.return_payment_url);
        // 异步回调
```

```
alipayRequest.setNotifyUrl(AlipayConfig.notify_payment_url); // 在
公共参数中设置回跳和通知地址

// 参数
// 声明一个map 集合
HashMap<String, Object> map = new HashMap<>();
map.put("out_trade_no", orderInfo.getOutTradeNo());
map.put("product_code", "FAST_INSTANT_TRADE_PAY");
map.put("total_amount", orderInfo.getTotalAmount());
map.put("subject", "test");
alipayRequest.setBizContent(JSON.toJSONString(map));
return alipayClient.pageExecute(alipayRequest).getBody(); //
调用 SDK 生成表单;
}
}
```

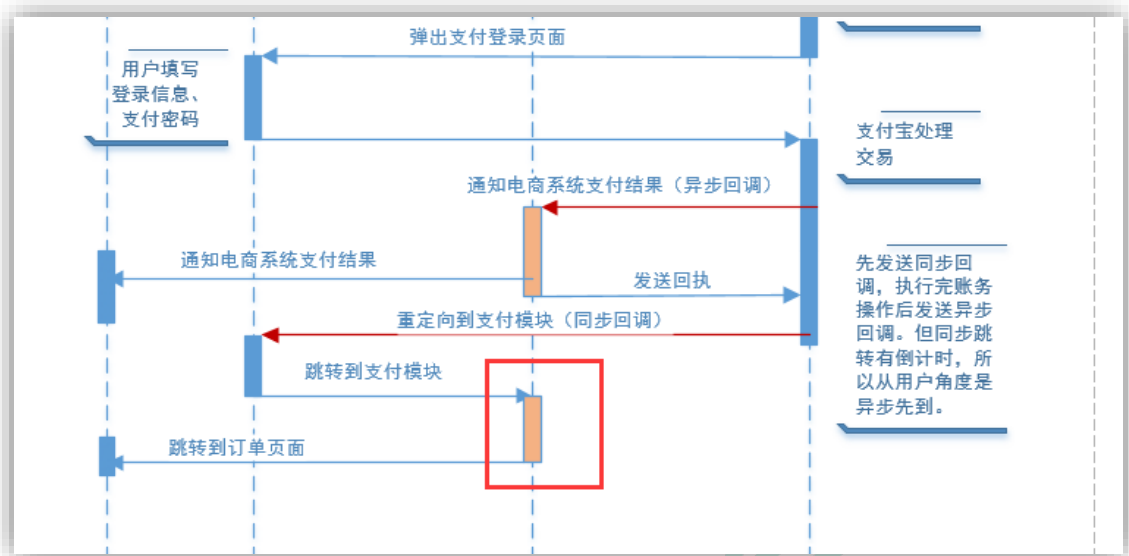
AlipayController

```
@Controller
@RequestMapping("/api/payment/alipay")
public class AlipayController {

    @Autowired
    private AlipayService alipayService;

    @RequestMapping("submit/{orderId}")
    @ResponseBody
    public String submitOrder(@PathVariable(value = "orderId") Long
orderId, HttpServletResponse response){
        String from = "";
        try {
            from = alipayService.createaliPay(orderId);
        } catch (AlipayApiException e) {
            e.printStackTrace();
        }
        return from;
    }
}
```

三、支付后回调—同步回调



AlipayController

```
/**
 * 支付宝回调
 * @return
 */
@RequestMapping("callback/return")
public String callBack() {
    // 同步回调给用户展示信息
    return "redirect:" + AlipayConfig.return_order_url;
}
```

这里 requestMapping 对应的路径必须与之前传给支付宝的 `AlipayConfig.return_order_url` 保持一致。

```
sequenceDiagram
    participant User
    participant PaymentModule
    participant MerchantSystem
    participant AlipaySystem

    User->>PaymentModule: 用户填写登录信息、支付密码
    PaymentModule->>AlipaySystem: 弹出支付登录页面
    AlipaySystem->>PaymentModule: 支付宝处理交易
    PaymentModule->>MerchantSystem: 通知电商系统支付结果
    MerchantSystem->>PaymentModule: 通知电商系统支付结果（异步回调）
    PaymentModule->>AlipaySystem: 发送回执
    AlipaySystem->>PaymentModule: 重定向到支付模块（同步回调）
    PaymentModule->>User: 跳转到支付模块
    User->>PaymentModule: 跳转到订单页面
```

用户填写登录信息、支付密码

弹出支付登录页面

支付宝处理交易

通知电商系统支付结果（异步回调）

通知电商系统支付结果

发送回执

重定向到支付模块（同步回调）

跳转到支付模块

跳转到订单页面

先发送同步回调，执行完账务操作后发送异步回调。但同步跳转有倒计时，所以从用户角度是异步先到。

确认并记录用户已付款，通知电商模块。新版本的支付接口已经取消了同步回调的支付结果传递。所以用户付款成功与否全看异步回调。

- 1、验证回调信息的真伪
- 2、验证用户付款的成功与否
- 3、把新的支付状态写入支付信息表{paymentInfo}中。
- 4、通知电商其他模块
- 5、给支付宝返回回执。

```
/**
 * 支付宝异步回调 必须使用内网穿透
 * @param paramMap
 * @param request
 * @return
 */
```

```
@RequestMapping("callback/notify")

@ResponseBody
public String alipayNotify(@RequestParam Map<String, String>
paramMap) {
    System.out.println("回来了!");
    boolean signVerified = false; // 调用SDK 验证签名
    try {
        signVerified = AlipaySignature.rsaCheckV1(paramMap,
AlipayConfig.alipay_public_key,
AlipayConfig.sign_type);
    } catch (AlipayApiException e) {
        e.printStackTrace();
    }
    // 交易状态
    String trade_status = paramMap.get("trade_status");
    String out_trade_no = paramMap.get("out_trade_no");
    // true
    if (signVerified) {
        // TODO 验签成功后，按照支付结果异步通知中的描述，对支付结果
        中的业务内容进行二次校验，校验成功后在 response 中返回 success 并继续商户自
        身业务处理，校验失败返回 failure
        if ("TRADE_SUCCESS".equals(trade_status) ||
"TRADE_FINISHED".equals(trade_status)) {
            // 但是，如果交易记录表中 PAID 或者 CLOSE 获取交易记录
            中的支付状态 通过 outTradeNo 来查询数据
            // select * from paymentInfo where out_trade_no=?
            PaymentInfo paymentInfo =
paymentService.getPaymentInfo(out_trade_no,
PaymentType.ALIPAY.name());
            if (paymentInfo.getPaymentStatus() ==
PaymentStatus.PAID.name() || paymentInfo.getPaymentStatus() ==
PaymentStatus.CLOSED.name()) {
                return "failure";
            }
            // 正常的支付成功，我们应该更新交易记录状态
            paymentService.paySuccess(out_trade_no, PaymentType.ALIPAY.name(),
paramMap);
            return "success";
        }
    } else {
        // TODO 验签失败则记录异常日志，并在 response 中返回 failure.
        return "failure";
    }
    return "failure";
}
```

PaymentService

实现类

```
@Override
public PaymentInfo getPaymentInfo(String out_trade_no, String
name) {
    QueryWrapper<PaymentInfo> paymentInfoQueryWrapper = new
QueryWrapper<>();
    paymentInfoQueryWrapper.eq("out_trade_no", outTradeNo
).eq("payment_type", paymentType);
    PaymentInfo paymentInfo =
paymentInfoMapper.selectOne(paymentInfoQueryWrapper);
    return paymentInfo;
}

@Override
public void paySuccess(String outTradeNo, String paymentType,
Map<String, String> paramMap) {
    PaymentInfo paymentInfo = this.getPaymentInfo(outTradeNo,
paymentType);
    if (paymentInfo.getPaymentStatus() == PaymentStatus.PAID.name()
|| paymentInfo.getPaymentStatus() == PaymentStatus.CLOSED.name()) {
        return;
    }
    PaymentInfo paymentInfoUpd = new PaymentInfo();
    // update paymentInfo set PaymentStatus =
PaymentStatus.PAID , CallbackTime = new Date() where out_trade_no = ?
    paymentInfoUpd.setPaymentStatus(PaymentStatus.PAID.name());
    paymentInfoUpd.setCallbackTime(new Date());
    paymentInfoUpd.setCallbackContent(paramMap.toString());
    this.updatePaymentInfo(outTradeNo, paymentInfoUpd);
    // 表示交易成功！
}

@Override
public void updatePaymentInfo(String outTradeNo, PaymentInfo
paymentInfoUpd) {
    QueryWrapper<PaymentInfo> queryWrapper = new QueryWrapper<>();
    queryWrapper.eq("out_trade_no", outTradeNo);
    paymentInfoMapper.update(paymentInfoUpd, queryWrapper);
}
```

五、退款

直接在浏览器发起请求即可！

<http://payment.gmall.com/refund?orderId=98>

商户与客户协商一致的情况下，才可以退款！

```
// 发起退款！订单编号 http://payment.gmall.com/refund?orderId=118
@RequestMapping("refund/{orderId}")

@ResponseBody
public Result refund(@PathVariable(value = "orderId")Long orderId) {
    // 调用退款接口
    boolean flag = alipayService.refund(orderId);
    return Result.ok(flag);
}

boolean refund(String orderId);

@Override
public boolean refund(Long orderId) {

    AlipayTradeRefundRequest request = new
    AlipayTradeRefundRequest();

    OrderInfo orderInfo = orderFeignClient.getOrderInfo(orderId);
    HashMap<String, Object> map = new HashMap<>();
    map.put("out_trade_no", orderInfo.getOutTradeNo());
    map.put("refund_amount", orderInfo.getTotalAmount());
    map.put("refund_reason", "颜色浅了点");
    // out_request_no
    request.setBizContent(JSON.toJSONString(map));
    AlipayTradeRefundResponse response = null;
    try {
        response = alipayClient.execute(request);
    } catch (AlipayApiException e) {
        e.printStackTrace();
    }
    if (response.isSuccess()) {
        // 更新交易记录：关闭
        PaymentInfo paymentInfo = new PaymentInfo();
        paymentInfo.setPaymentStatus(PaymentStatus.CLOSED.name());
        paymentService.updatePaymentInfo(orderInfo.getOutTradeNo(),
        paymentInfo);
        return true;
    } else {
        return false;
    }
}
```



```
}
```

六、常见面试问题：

1、支付宝支付需要的参数？

公共的:appid 私钥 公钥 签名方式 字符集 请求数据类型

请求支付接口: 订单交易编号 金额 产品销售码 标题

退款: 金额、交易编号（支付宝那边的交易号）或者订单交易编号

2、支付锁库存吗？

不锁，支付收钱呢，不要和其他业务耦合。

3、支付日志表（信息表）记录了什么东西？（主要是看表里的字段）

订单 id 和交易编号、支付类型、金额、用户 id、回调时间 回调内容

作用：对账。