# 支付流程

## 1.确认商户;

商户与通道,相互透明, 在支付实际请求中, 均以子账号形式请求( 依赖与通道是否支持, 不支持则自建用户信息管理, 并在提交支付的每个阶段记录流水与支付状态) , 在清算日按按商户对应的子账号进行清算 ( 我们与通道清算之后, 再分别计算每个子账号的金额, 初期可能人为计算 )

商户的配置 , 包含但不限于相关资料 , 开通的并使用的通道 ( 基于各个通道的情况不一致, 有的通道更敏感 , 选择性的开通更好 )

## 确认场景;

线上支付场景一般由用户的支付终端来区分 : web ( 电脑网页 ) ,wap ( 手机网页 ) , App ( ios , 安卓 ) ; 每种支付场景都用不同的用户支付流程 ( 通道调用也不是完全一直, 各通道本身也是各不相同 ) , 为了通道的统一调用, 不仅需要对外提供一致的api调用,方便商户的对接, 在系统内部实现的各个通道接口,也应该使用统一的调用规则 ( 在调用通道前多一层调用封装也是可选方案 ) , 这样才能在做多通道的自动切换时不影响其他的业务 ( 日志 , 警报触发等 ) .

## 确认可用通道与本次请求的使用通道 ( 依赖”商户”,”场景” );

每个支付请求, 都需要区分是有那个子账号 ( 商户 ) 发起的 , 并在当前的支付场景分类下( 安卓设配 和 pc 显然支付流程是不一致的 )的可用通道中选择其一进行后续支付 ( 需保正,任意支付通到的用户体验是完全一致的 ) , 这也是做通道切换的基础;

## 得到通道配置 ( 依赖”多通道选择” );

通道选择的策略, 子账号 ( 商户 )必须是开通的对应通道( 这里的开通是指人为审核 ), 按照预定策略选择 ( 伐值型 , 权重型 , 业务型 , 复合型等 ) , 每种都应有单独的数据库关系来表示, 而最终的选策略 ( 复合型 ) , 则应由更多一张数据表表示;

## 使用对应通道的通道账户支付;

由于每个通道各自的风险评估与签订协议的不同, 使用子账号支付, 是否被通道支持的情况也是不统一的, 所以在此处同样需要设计, 使得账户选择在多中情况下能够得到统一的调用, 这也是清算的基础, 即使子账号支付没有得到通道的对策支持, 每笔数据也都是重要的数据, 对账也是必要的

## 使用通道对应的支付加密;

由于各个通道, 所用的加密方案各不相同, 甚至大相径庭, 所以每种通道的加密方式都使用最大兼容性的代码写法, 例如 : 加密需要使用的证书文件, 在A通道提供的demo中是以路径文件名的形式调用, 而B通道却是需要传入证书文件的内容, 因此我们在保存密钥证书的时候都选择文件存储的方式, 而非数据库保存文件内容, 一使得配置信息的存储的统一, 也使调用得以方便.

## 用户支付, 用户相关操作 ( 不同场景用户的确认方式是不同的 );

用户支付后一般是在通道的页面或者app下进行的( 也不全是 ). 这部分一般只要调用的通道的支付场景不出错, 就不会有什么问题

## 通道的支付结果回调确认;

验证回调是否来自正确的通道是非常重要的事情, 但也是最可能在进度压力下的忽略的.

## 通知商户支付结果;

同步与异步通知商户结果, 要保证每个商户都有不同的请求验证.

同样需要设计可以让商户的知道他们收到的请求是否是我们发出的, 主动调用接口.