# 商城购买流程

这里有可能是虚假信息

用户点击钻石某档

App判断该商品是否处于销售状态

不属于销售状态

退出流程

处于销售状态，发送请求获得购买号（参考1）

App将购买号等本地缓存,发送苹果购买请求（本地状态：开始购买）

Appstore 处理购买请求。返回交易信息

交易状态判断（本地状态：购买失败-后台未确认；购买成功-后台未验证）参考2

后台根据购买号，结束购买状态（参考2）

购买失败-后台未确认

本地状态：购买失败-后台确认

获取苹果购买成功交易凭证

购买成功-后台未验证（获取交易凭证）

将base64好后的交易凭证上传后台

分别验证购买号，以及苹果订单号的状态（参考3）

将交易凭证发送给苹果验证

苹果接收到验证请求，返回json数据

后台接收返回的json数据

确定用户购买的内容，验证交易状态，修改状态（验证通过修改状态：假购买|真购买|已购买）

真购买才加钻石，真购买条件state=0&&bundle\_id&&product\_id&&transaction\_id唯一

返回数据（完成DP-YZ9接口数据返回）

修改本地购买状态

购买成功上传凭证（DP-YZ9接口）



后台接口：

1：商品处于销售状态，上传后台缓存，开始流程=====上传参数:action,用户id,token,shopid(购买的商品id).bundleid(包名)。后台需要返回一个购买号，这个号是唯一的，与这次交易是绑定的。（服务器应该有一个购买状态字段）

2：购买失败=====上传参数：action，用户id，token,购买号。 后台结束购买状态

3：购买成功====上传参数：action,用户id,token,购买号，apptransactionid(返回的订单号)，交易凭证（这个凭证是已经处理好的，可以直接请求苹果（请求方式post））。

交易凭证请求Url有2个：1：<https://buy.itunes.apple.com/verifyReceipt>（这个是正式URL） 2：<https://sandbox.itunes.apple.com/verifyReceipt>（这个是沙盒测试URL）。这里的处理是先验证正式的URL,如果返回的status==21007，那么在请求测试URL。返回的status==0则表示验证成功。解析Json得到返回的bundle\_id,product\_id,transaction\_id.=========通过比较bundle\_id == bundleid && product\_id == shopid && transaction\_id == apptransactionid （transaction\_id需要在数据库中不存在）相同才为用户添加钻石。

If( )

{

<https://buy.itunes.apple.com/verifyReceipt>

}

else

{

<https://sandbox.itunes.apple.com/verifyReceipt>

}

result = post 请求

if (result)//不为空

{

dict = 解析json

if (dict)//dict不为空

{

if (status == 21007)//这个是沙盒测试账号请求了正式的URL

{

请求测试URL

}

else if (status == 0)//验证成功

{

比较bundle\_id == bundleid && product\_id == shopid && transaction\_id == apptransactionid

}

}

}

4:某些情况下，有可能在第2次才会上传凭证，订单号等，而第2次又不是以前登录的那个账号，这种情况下，还是要将以前的用户的购买信息上传。这里需要一个单独的购买成功-未验证的上传接口。（或者将这个接口与3的接口合并为1个）