乘车码接入规范 V4.0.1(通用 SDK 接入)

腾讯科技（深圳）有限公司

文档历史

修订日期	修订内容	修订版本
2016-12-30	创建	1.0.0
2017-08-16	方案修改	2.0.0
2018-03-10	1：增加出账单下载接口、查询卡 id 接口 2：修改沙箱接入流程解释 3：增加接口使用示例 4：增加服务上线流程 5：修改商户支持项解释	3.0.0
2019-05-29	1：机具 SDK 新增防重扫刷 2：新增机具 SDK 初始化接口 3：密钥下载接口包含公钥和 mac 根密钥 4：账务接口更新统一使用 version=3.0	4.0.0(通用 SDK)
2019-06-18	1：新增账单 4.0 版本 version=4.0	4.0.1(通用 SDK)

目录

文档历史 1
目录 2
第 1 章 引言 4
1.1. 概述 4
1.2. 阅读范围 4
1.3. 名词解析 4
1.4. 文档约定 4
1.5. 接入前需准备的资料 5
1.5.1. 合作伙伴需提供的资源 5
1.5.2. 腾讯提供的资源 5
第 2 章 总体方案 6
2.1 业务流程 6
2.1.1 总体流程 6
2.1.2 二维码扫码交互 7
2.1.3 定期密钥更新 8
2.1.4 定期黑名单更新 8
2.2 数据设计 9
2.2.1 二维码结构 9
2.2.2 合作伙伴机具的二维码验证流程 10
第 3 章 协议说明 11
3.1 机具 SDK 11
3.1.1 机具 SDK 功能和集成 11
3.1.2 腾讯机具 SDK 验证二维码流程图 12
3.1.3 SDK 初始化 13
3.1.4 二维码验证接口 14
3.1.5 机具 SDK 错误码说明 15
3.2 后台 SDK(java) 16
3.2.1 SDK 集成说明 16
3.2.2 SDK 初始化 16
3.2.3 密钥列表下载接口 17
3.2.4 黑名单下载接口 19
3.2.5 申请扣款接口 20
3.2.6 查询扣款接口 24
3.2.7 申请退款接口 26
3.2.8 查询退款接口 28
3.2.9 下载对账单接口 29
3.2.10 下载资金账单接口 31
3.2.11 推送行程数据接口(可选) 35
3.2.12 单边补登消息通知接口(可选) 37
3.2.13 手机号查询卡 id 接口 39

第 4 章 沙箱联调环境 41
4.1 说明 41
4.2 接入方法 41
第 5 章 商户接入要求 42
5.1 机具功能 42
5.2 机具硬件要求 42
5.3 服务器要求 42
5.4 运维注意事项 42
第 6 章 商户后台能力要求 43
6.1 后台能力要求 43
6.2 后台能力分值统计 43
第 7 章 正式上线流程 45
7.1 上线流程介绍 45
7.1.1 腾讯后台发布与线上环境测试 45
7.1.2 产品体验与功能完整性检查 45
7.1.3 开放入口 45
7.2 通信密钥生成说明 45
第 8 章 常见问题 47
第 9 章 附件 48
9.1 附录 A 48
9.1.1 订单类型场景 48
9.1.2 后台 SDK 错误码 49
9.1.3 订单异常拦截 49
9.2 机具硬件配置表 50
9.3 机具扫码语音及文案建议 50
9.4 腾讯乘车码机具功能检查表 51
9.4.1 机具功能验收表 51
9.4.2 机具体验验收表 51
9.5 腾讯乘车码后台检查表 52
9.6 银行列表 53

第1章 引言

1.1. 概述

此规范主要从乘车码业务流程、二维码结构、接口、联调环境、商户接入要求、商户后台能力要求、上线流程等各个方面介绍腾讯乘车码的原理和接入方法，帮助合作伙伴快速掌握接入腾讯乘车码所需掌握的基本内容。

此规范的解释权归腾讯公司唯一所有。

1.2. 阅读范围

从事腾讯乘车码接入相关的开发者，审阅者，维护和测试人员等。

1.3. 名词解析

术语	说明
腾讯乘车码	腾讯和合作伙伴联合发布的，用户用来乘坐公交或地铁等交通工具的二维码，包含用户身份 ID、支付方式、时间戳、以及安全校验信息等，可以被机具快速读取。
商户	腾讯乘车码的合作伙伴，商户号主体。
行业数据	由合作伙伴后台生成且返回给腾讯后台，作为二维码数据的一部分，包括了合作伙伴公交或者地铁行业相关的数据，用于合作伙伴验证。
卡证书	腾讯后台生成且由腾讯私钥签名的二维码静态数据，主要包括用户 open_id、二维码有效时长、算法标识等数据。
公钥列表	通过腾讯开放平台下载并下发到合作伙伴机具，是用于对二维码卡证书数据做签名验证的公钥列表，以防止二维码静态数据被非法篡改。
MAC 根密钥	通过腾讯开放平台下载并下发到合作伙伴机具，是用于对二维码整体数据（包括行业数据、卡证书和动态数据等）做签名的根密钥，系统以此根密钥分散得到子密钥进行签名，以防止二维码数据被非法篡改。
黑名单	通过腾讯开放平台下载并下发到合作伙伴机具，用于机具对恶意用户进行拦截，该数据会定期更新。
对账单	通过腾讯开放平台下载，合作伙伴可用此对账单进行对账。
机具 SDK	由腾讯提供，用于解析二维码数据和验证二维码的合法性和有效性，合作伙伴基于该 SDK 编写机具程序。
后台 SDK	由腾讯提供，合作伙伴可通过该 SDK 接口与腾讯开放平台进行对接通信，便于合作伙伴快速接入。
腾讯开放平台	指腾讯提供给合作伙伴访问乘车码相关后台服务的平台，这里特指 https://open-wlx.tenpay.com 提供的相关后台接口和服务。
ykt_id	指腾讯分发给商户的一卡通 id。

1.4. 文档约定

在接口参数的描述中，使用以下术语约定	
存在	说明
M	must 必须存在
O	optional 可选参数
C	conditional 在某些条件下必
R	copy from request 与请求串一致
-	meaningless 无意义

1.5. 接入前需准备的资料

1.5.1. 合作伙伴需提供的资源

1. 《机具硬件配置表》（模版见附件 9.2）
2. 上线城市乘车单笔最高票价

注：请按照腾讯模板填写并提交至腾讯乘车码接入团队。

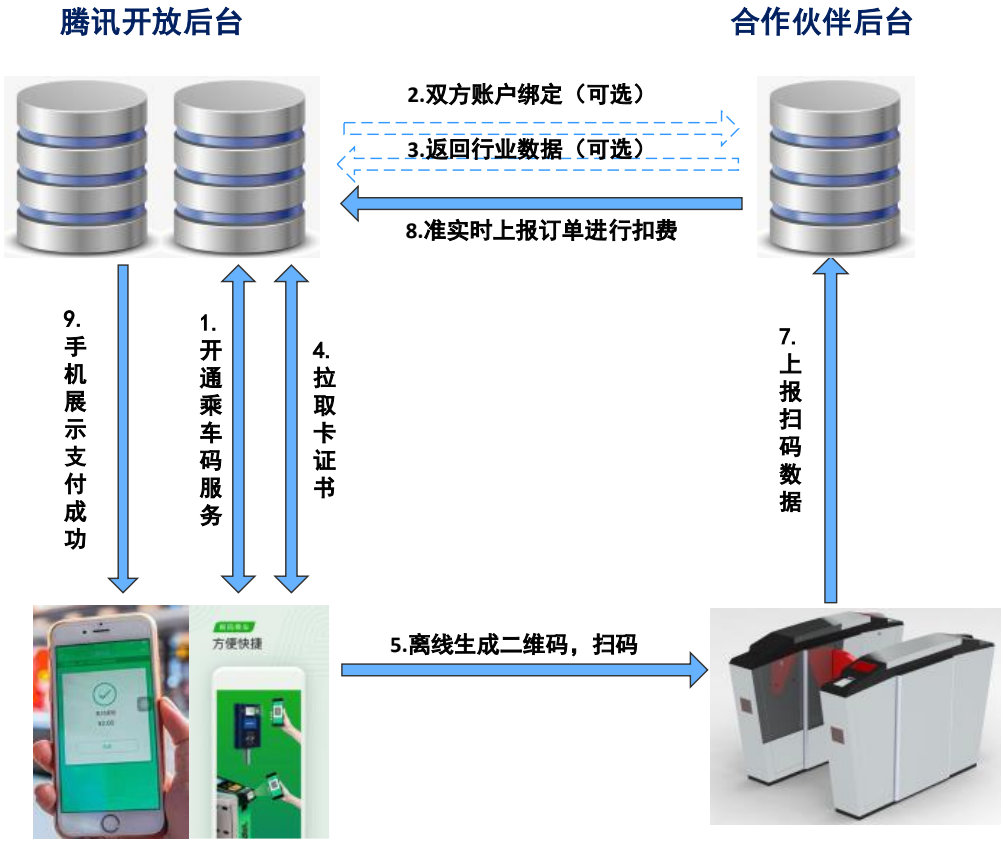
1.5.2. 腾讯提供的资源

1. 《腾讯乘车码接入规范》（即本文档）
2. 机具 SDK
3. 后台 SDK（可选）

第2章 总体方案

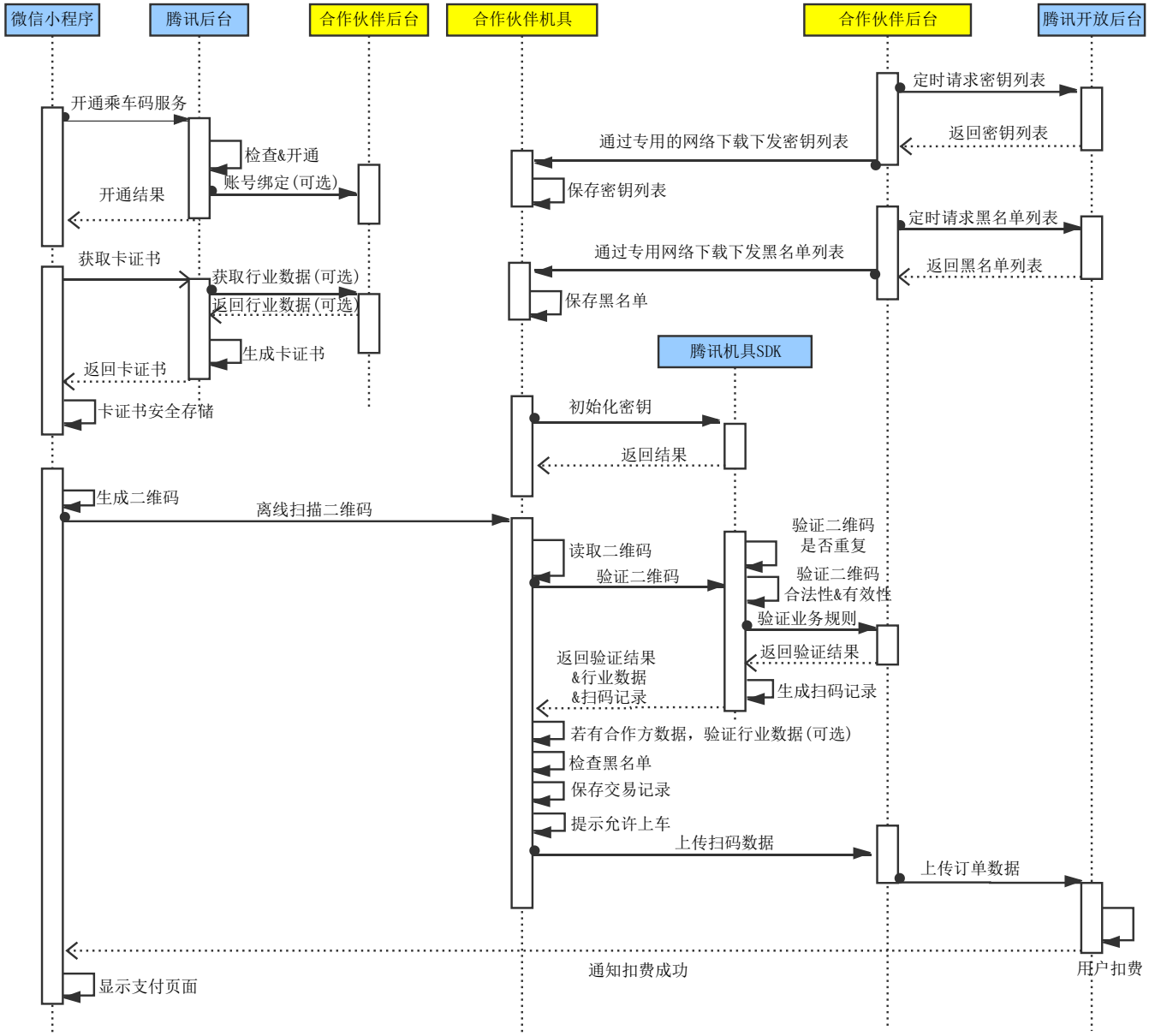
2.1 业务流程

2.1.1 总体流程

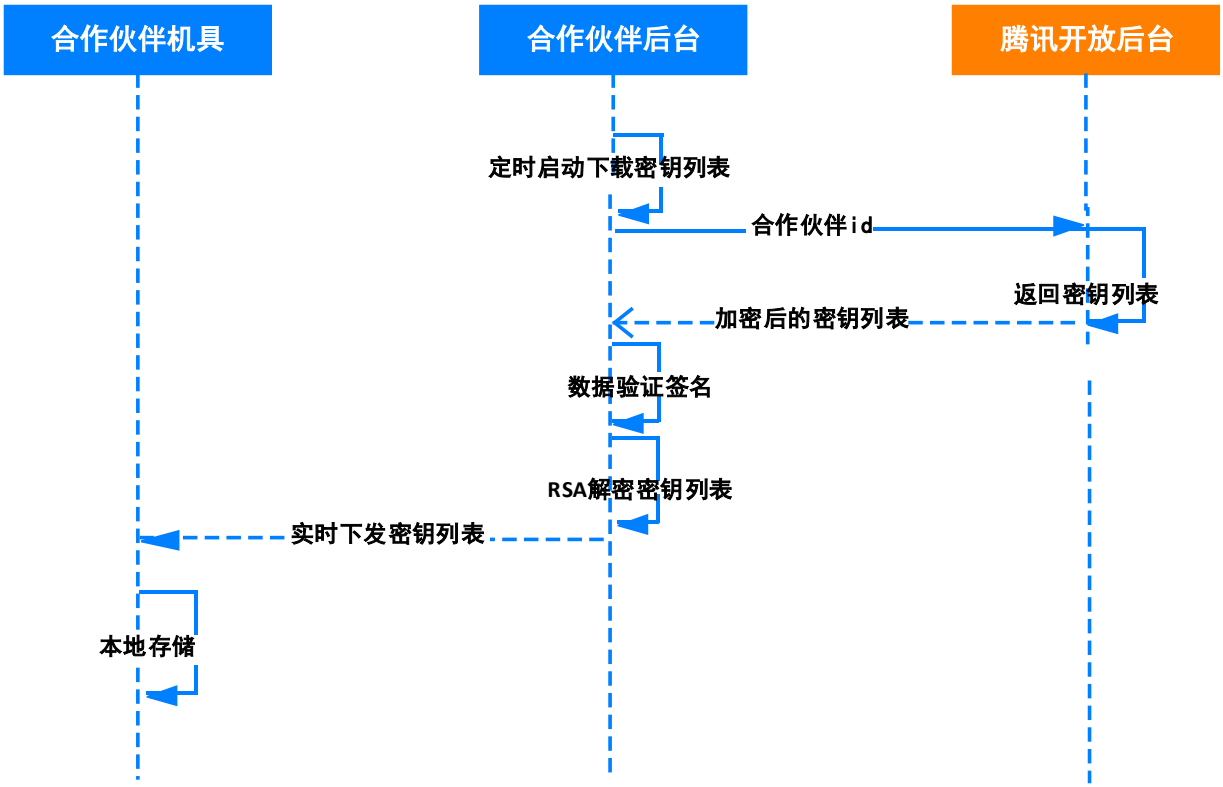


1. 用户打开微信“腾讯乘车码”小程序，开通腾讯乘车码服务；
2. （可选）腾讯后台和合作伙伴后台进行双方账户绑定；
3. （可选）若合作伙伴需要把行业数据植入二维码，则用户开通后返回行业数据给腾讯后台；
4. 用户拉取卡证书用于离线生成二维码，该卡证书作为离线生成二维码的唯一凭证，微信小程序加密保存在用户手机；
5. 手机离线生成二维码后，用户在机具上扫码乘车；
6. 机具调用腾讯提供的机具 SDK 对二维码合法性和有效性进行验证，若联合发码，合作伙伴可自行对行业数据有效性进行验证，全部验证通过后，则为有效二维码；
7. 机具准实时单笔或批量上报扫码数据到合作伙伴后台；
8. 合作伙伴后台准实时单笔或批量上报订单数据到腾讯后台进行扣款；
9. 腾讯后台收到订单后，若扣款成功，则下发支付成功消息到用户手机，微信小程序展示支付成功页；

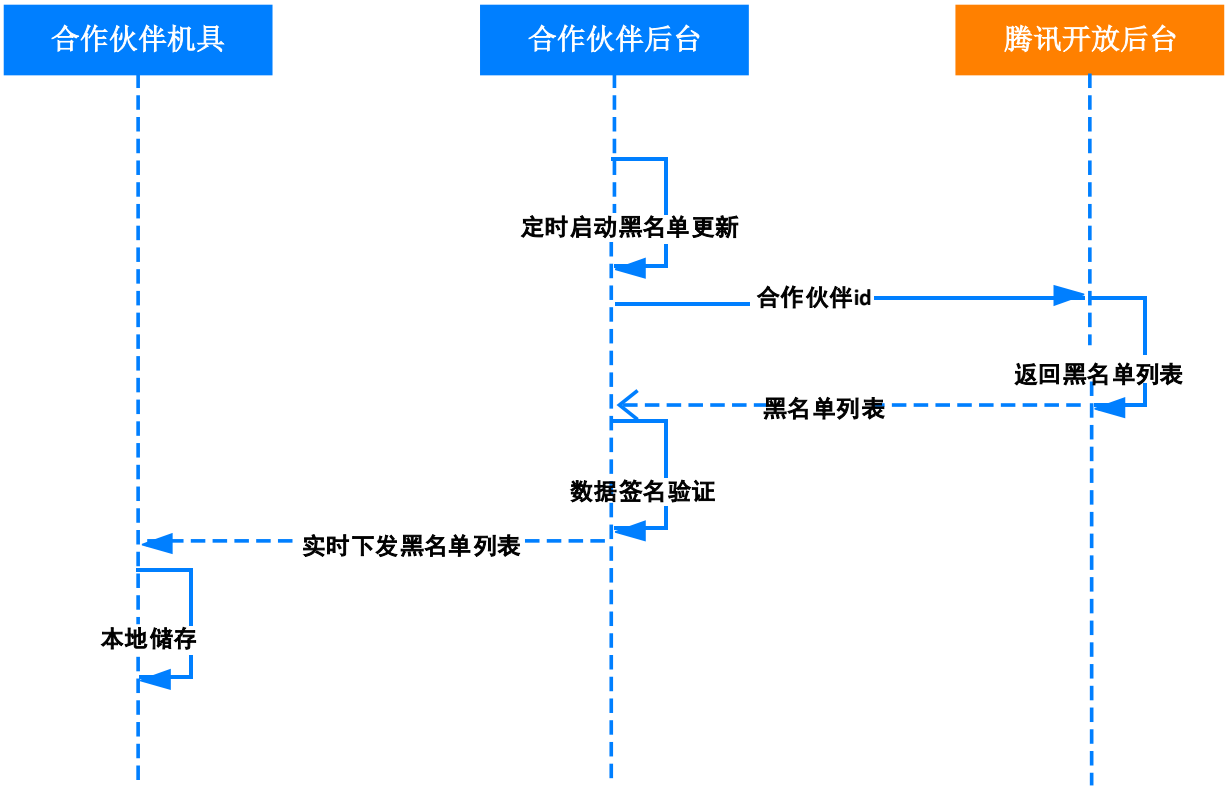
2.1.2 二维码扫码交互



2.1.3 定期密钥更新



2.1.4 定期黑名单更新



2.2 数据设计

2.2.1 二维码结构

二维码由标识码和 base64 数据体构成，标识码用于终端快速过滤非本规范二维码，其结构如下所示。

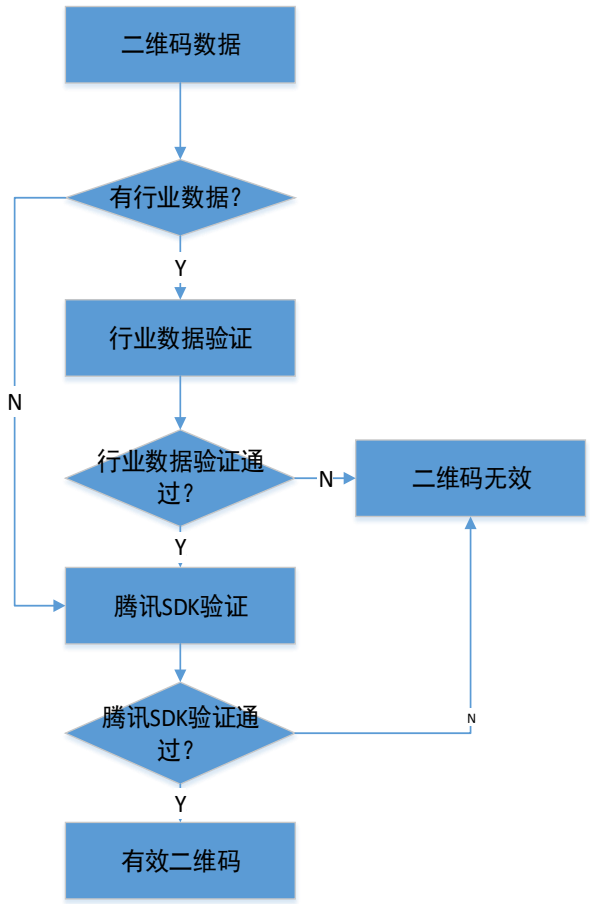
拆解部分	内容	长度（字节）	类型	说明
应用标识	应用标识	2	char	“TX”
base64(二维码正文)：以下为二维码正文				

二维码正文

拆解部分	内容	长度（字节）	类型	说明
行业数据	行业数据长度	2	BIN	行业数据长度
	行业数据	N	BIN	行业数据
卡证书数据	卡证书版本	1	BIN	卡证书版本 对应终端规则版本， 当前版本为 0x01
	卡证书签名算法 ID	1	BIN	0x05：ECC 224R1
	动态数据签名算法 ID	1	BIN	0x00：MAC 分散密钥方式
	卡证书公钥索引	1	BIN	闸机验卡证书签名的公钥列表索引
	appid	4	BIN	合作伙伴 ID，腾讯分配
	openid	8	BIN	标识用户账户的编号
	卡证书签发时间	4	BIN	UNIX 时间值（表示从 1970-01-01 00:00:00 到卡证书签发相差的时间，单位是秒），高字节在前
	卡证书失效时间	4	BIN	UNIX 时间值（表示从 1970-01-01 00:00:00 到卡证书签发相差的时间，单位是秒），高字节在前
	二维码失效时间	2	BIN	二维码展示的有效时间，单位秒，高字节在前
	单笔限额	4	BIN	单位分
	用户标签	2	BIN	用户标签，优惠用
	保留	4	BIN	（第一个 BIT 为 1 表示充值模式， 0 为非充值模式）
	动态密钥信息	变长	BIN	变长，跟“动态数据签名算法 ID”字段配套使用
	卡证书签名数据	变长	BIN	对卡证书数据+行业数据签名
动态数据	支付类型	1	BIN	由账户发行方自定义

	二维码生成时间	4	BIN	Unix 时间戳（表示从 1970-01-01 00:00:00 到卡证书签发相差的时间，单位是秒），高字节在前
	TAC	4	BIN	用户 TAC 签名
	保留	4		保留 前两个字节是 0 （后两个字节表示余额 单位是分）
二维码签名	二维码数据签名	N	BIN	根据动态数据签名算法 ID，对除“TX”外的二维码所有数据进行签名的结果

2.2.2 合作伙伴机具的二维码验证流程



合作伙伴可以自行验证行业数据。只有行业数据验证通过，同时腾讯机具 SDK 也验证通过，才算二维码验证完成。

第3章 协议说明

3.1 机具 SDK

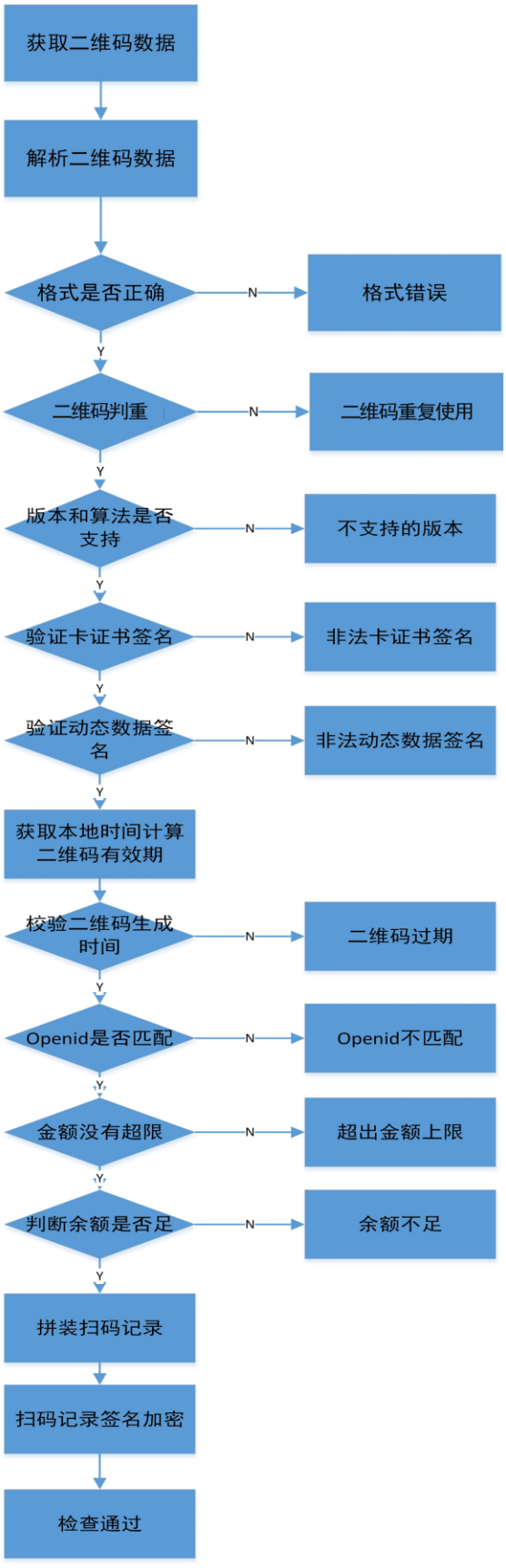
3.1.1 机具 SDK 功能和集成

1. 机具 SDK 提供以下功能

功能接口	接口简述
SDK 初始化接口	将密钥列表初始化到机具 SDK 中
二维码验证接口	验证二维码的合法性和有效性，若验证通过，该接口会返回腾讯签名的扫码记录。

2. 机具 SDK 集成
   1) 机具 SDK 包，包括 sdk.h，libtxccm_pos_sdk.so
   2) 在机具程序要加载 libtxccm_pos_sdk.so，通过 sdk.h 定义的函数，初始化接口并验证二维码合法性。
3. 使用说明
   3.1 打开动态链接库 libtxccm_pos_sdk.so
   1) 包含头文件
      #include <dlfcn.h>
   2) dlopen 打开指定的动态链接库
      void * dlopen(const char *pathname, int mode)
   3) dlsym 获取接口函数
      void* dlsym(void* handle,const char* symbol)
   4) 编译时候要加入 -ldl (指定 dl 库)
      注: 示例请参考 DEMO 中的 sdkmain.c 文件
   3.2 引用 cJSON.c 和 cJSON.h
   利用 cJSON 库以 json 的形式来组成一个字符串，编译时候要加入 -lm (指定数学 math 库)
4. 机具负责具体磁盘存储和网络访问

3.1.2 腾讯机具 SDK 验证二维码流程图



3.1.3 SDK 初始化

接口描述

闸机通过传入密钥列表调用本接口完成腾讯机具 SDK 初始化。本接口在闸机每次开机时调用一次即可。

接口函数

```
int init_txccm_pos_sdk(const char* key_list)
```

输入参数

字段名	类型	必填	说明
key_list	const char*	M	json 形式的密钥列表

密钥列表说明

key_list:

字段名	类型	必填	说明
key_id	int	M	密钥 id
key_type	int	M	密钥类型
key_value	char*	M	密钥内容

key_list 示例：

```
[
    {
        "key_id":1,
        "key_type": 1,
        "key_value":"04535F8AF4CE0B24AB3F6D66D9C662764A994"
    },
    {
        "key_id":2,
        "key_type": 2,
        "key_value":"B+AF6vcFoEccX8Gi4v9xHTighv5QlND3pDqtmU"
    }
]
```

输出参数

无

错误码

请参见 3.1.5 错误码说明

接口示例

见 sdkmain.c 中的 sdk_init 函数。

3.1.4 二维码验证接口

接口描述：

验证二维码的合法性和有效性，若验证通过，该接口会返回腾讯签名的扫码记录，合作伙伴需要把该扫码记录作为扫码数据的一部分上传到后台，并且通过订单一起推送到腾讯后台作为验证凭证。

接口函数：

```
int txccm_verify_qrcode(TXCCM_REQUEST* txccm_request, TXCCM_RESPONSE* txccm_response)
```

输入参数：

字段名	类型	说明
txccm_request	TXCCM_REQUEST	验证二维码，输入参数，封装内容见结构体

输出参数：

字段名	类型	说明
txccm_response	TXCCM_RESPONSE	验证二维码，输出参数，封装内容见结构体

结构体说明

TXCCM_REQUEST

字段名	类型	说明
qrcode	char *	二维码
qrcode_len	int	二维码长度
pos_param	char *	json 形式的闸机信息
以下是闸机信息 pos_param 的具体内容：		
payfee	int	实际扣款金额（扣除优惠），单位是分
scene	int	场景： 1 一次扫码计费， 2 多段扫码计费
scantype	int	扫码类型：1 进站 2 出站
scan_time	int	扫码时间
pos_id	char[64]	机具 ID，长度 10-64
pos_trx_id	char[64]	机具流水号，长度 10-64

TXCCM_RESPONSE

通过 SDK_QRCODE_OUT_New()， SDK_QRCODE_OUT_Free()来管理内存

字段名	类型	说明
card_id	char[32]	用户卡号

ykt_id	unsigned int	商户号
max_pay_fee	unsigned int	单笔限额 单位分
biz_data_len	int	合作方数据长度
biz_data	unsigned char[256]	合作方数据
record	char *	脱机交易记录 机具记录在本地，异步上传
record_len	int	脱机交易记录长度

注：所有错误保存情况发生时保存原始二维码值和错误日志。

错误码

请参见 3.1.5 错误码说明

接口示例

见 sdkmain.c 中的 ccm_verify 函数。

3.1.5 机具 SDK 错误码说明

返回值	错误码描述	开发处理方式
0	校验通过	允许乘车
-10000	二维码格式错误	提示格式错误无法解析
-10001	卡证书公钥错误	下载的卡证书公钥有误
-10002	卡证书验签失败	提示卡证书非法
-10003	卡证书用户公钥错误	提示卡证书数据错误
-10004	二维码验签错误	提示非法二维码
-10005	卡证书过期	提示用户联网更新
-10006	闸机扫码时间大于二维码有效期	提示用户确认手机时间并刷新二维码
-10007	超过最大金额	提示金额超限制
-10008	保留	
-10009	输入的 card_id 不匹配	通过输出参数可以获取正确的 card_id
-10010	参数错误	提示格式错误无法解析
-10011	内存申请错误	保持足够的运行内存
-10012	卡证书签名算法不支持	
-10013	加密的 mac 根密钥解密失败	
-10014	mac 校验失败	
-10015	二维码签名算法不支持	
-10016	扫码记录加密失败	
-10017	扫码记录编码失败	
-10018	闸机扫码时间小于二维码生成时间	提示用户确认手机时间并刷新二维码
-10021	输入的密钥列表为空	请确认机具 SDK 初始化接口的正确性
-10022	输入的密钥列表长度错误	请确认机具 SDK 初始化接口的正确性
-10023	密钥列表未进行初始化	请确认是否进行机具 SDK 初始化接口
-10024	密钥列表格式错误	请确认机具 SDK 初始化接口的正确性
-10025	密钥列表个数错误	请确认机具 SDK 初始化接口的正确性
-10026	输入的密钥列表中类型错误	请确认机具 SDK 初始化接口的正确性
-10031	输入的公钥列表中的字段名称错误	请确认公钥列表的正确性
-10032	输入的 MAC 列表中的字段名称错误	请确认 MAC 列表的正确性
-10033	公钥列表中无与二维码公钥 id 匹配的公钥	请确认公钥列表的正确性

-10034	MAC 列表中无与二维码 MACid 匹配的 MAC	请确认 MAC 列表的正确性
-10035	相同二维码重复扫码错误	请确认二维码是否重复
-20000	其它错误	

注：所有错误保存情况发生时保存原始二维码值和错误日志。

3.2 后台 SDK(java)

3.2.1 SDK 集成说明

	具体描述
BIN	txccm_server_sdk.jar，通过 SRC 目录下源码编译的 jar 包
DEMO	CcmBackendSDK.java，调用接口使用的 demo 文件
乘车码接入规范 V4.0.1(通用 SDK 接入)	从乘车码后台 SDK 集成功能、接口说明和联调环境三个方面介绍了乘车码 SDK 功能和使用方式

2. 功能集成列表

后台 SDK 集成了以下接口：

功能接口	接口简述
SDK 初始化接口	SDK 接口调用前数据初始化
密钥列表下载接口	下载安全密钥，用于机具验证二维码的有效性。
黑名单下载接口	下载黑名单用户列表，用于机具对恶意用户拦截。
申请扣款请求接口	传入用户乘车订单，对用户的乘车行程进行扣费。
查询扣款单接口	查询订单当前状态。
申请退款接口	对已经扣费成功的异常订单进行退款。
查询退款接口	查询退款订单当前状态。
下载对账单接口	下载历史扣款请求清单，进行对账，对因为网络异常等其他原因没有正常扣费的订单，可以通过扣款请求接口进行扣费。
下载资金账单接口	下载资金变动清单。
推送行程数据接口	腾讯后台推送进站信息到用户侧提醒用户已进站。
单边补登消息通知接口	分段计费的情况下，补登接口可将单边乘车记录补全出入站
手机号查询卡 id 接口	用户绑定的手机号查询腾讯乘车码对应的卡号

3.2.2 SDK 初始化

接口描述

商户与腾讯后台交互需要做签名验证，后台 SDK 初始化存储 ykt_id，腾讯侧的公钥，商户测的私钥。

接口函数

```
public CcmBackendSDK(String ykt_id, String server_pubkey, String client_prikey)
```

输入参数

字段名	类型	长度	必填	说明
ykt_id	string	8	M	商户号
server_pubkey	string		M	通讯密钥，腾讯侧提供的公钥
client_prikey	string		M	通讯密钥，商户侧的私钥

输出参数

无

错误码

无

接口示例

```
//SDK 初始化
String ykt_id="20000007";
String server_pubkey="MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC/Vy69K4ELwLL/wdhtLpff54I1\n" +
        "EOdzqNxRNy3BY8DFKuKDCS7AZUFUvPCKxbnUkAVJguft4eHscZQgvN3ipNW81+vu\n" +
        "iI2cRftmZpUbEti8g5wHZaNfqxQmTwwy5/xFuNN2U4eeiV3wo1nJ58tW4vsgrowR\n" +
        "EO5y8BoCeMNFAtaZdQIDAQAB";
String client_prikey="MIICXAIBAAKBgQCwg4s1heYuVHdeV9qsgXMJHO5XVlTxFVpxz/DgZlr/Wo/p2tzU\n" +
         "k3aJ33nVA0rAiwOWjyvNM5J1NbpKZRAV1qOPtsU0sRLKxMt+YrZF7RRoi5rJrZ8Z\n" +
         "LxPGdmLkbQXn7Qp3Jtn7iBPZkikiZc7w5yCNf/Xa54UbHFCUv5NGLXW+jwIDAQAB\n" +
         "AoGAOXkra6sEjtNL5rkeZ4rixPXZmTBwCeuU/nfhi39oY7q+Hzv3KXQ2ZaARUE15\n" +
         "GoZpDa3iajc/mdB7rtuHSEUSDh+6EtZCOFR8PZkOqunb7tCkvyjC6w0q0+jG7s5g\n" +
         "pEaySgmuKE8+kIZ0Re2as53zo2qfxzLkhP13XmhWro3tgkECQQDd//J/kilrIxfo\n" +
         "inmSFdtd8VFqUNEe8y1/TYIjZInIrhiDPNB/eWFNbt8R4qCeX15GhZ6ZUlOP0oEK\n" +
         "qypRkgovAkEAy4w0+xu0vZiu+GwgIob1izHEl7DA6vmCz9HiA5NY/cjGrQRgdmx6\n" +
         "DkbG7zC3fzeTnE1EWOckF6a2MeylGF1ZoQJALl+azl8/26t3ARJ8FrIOIu+X7Dd5\n" +
         "l5eAt4j/WFlWFt+XK0L24sn+M2innFrU5oBRdzXOTYTPA8obPplGu8df7QJBAKG3\n" +
         "2pwrbhU4ysM6/OkRuuKFfwFaAFxwMrs0sNJQbmLr8tWh5ZYRJ4RSPVnqpc+gc1m6\n" +
         "lfPgaO+Vl6ngr2bFPCECQFm0nyEMWKzaiTxfMf4lc0PsM3v6wNnAQAONF9AWN9Gl\n" +
         "IaydgmmZz+ZR65lQgFykghWf0/P+nzLdJ2fkqo8hoUI=";
 CcmBackendSDK ccmBackendSDK = new CcmBackendSDK(ykt_id,server_pubkey, client_prikey);
```

3.2.3 密钥列表下载接口

接口描述

密钥列表下载接口，密钥下载后安全下发到机具，用于机具 SDK 验证二维码的有效性。

接口函数

```
public synchronized KeyResult downloadKey()
```

输入参数

无

输出参数

字段名	类型	说明
keyResult	KeyResult	密钥列表下载结果

类说明

KeyResult:

字段名	类型	说明
retcode	int	错误码
retmsg	string	错误描述
以下字段在 retcode=0 时返回		
keylist	ArrayList<Keyinfo>	密钥列表

Keyinfo:

字段名	类型	说明
key_id	int	密钥 id
key_type	int	密钥类型
key_value	string	密钥内容

错误码

错误码	错误描述	解决方案
0	成功	
非 0	错误	联系腾讯开发对接人员或查看附录 9.1.2 后台 SDK 错误码(使用腾讯后台 SDK 的商户)

接口示例

```
1.  //*********************密钥列表下载接口*********************
2.  KeyResult keyResult = ccmBackendSDK.downloadKey();
3.  if(keyResult.retcode != 0)
4.  {
5.      Constant.LogDebug("retcode:"+keyResult.retcode+"\n");
6.      Constant.LogDebug("retmsg:"+keyResult.retmsg+"\n");
7.      Constant.LogDebug("密钥列表下载接口失败\n");
8.  }
9.  else {
10. //    key_type:1 为卡证书公钥，2 为 mac 密钥
11.     for(int i = 0;i<keyResult.keylist.size();i++) {
12.         Constant.LogDebug(Integer.toString(keyResult.keylist.get(i).key_id)+"\n");
13.         Constant.LogDebug(Integer.toString(keyResult.keylist.get(i).key_type)+"\n");
14.         Constant.LogDebug(keyResult.keylist.get(i).key_value+"\n");
15.     }
16.     Constant.LogDebug("密钥列表下载成功\n");
17.  }
```

使用说明：

需要商户后台定期（1 天）来拉取，商户后台更新秘钥数据并 push 到机具保存。

3.2.4 黑名单下载接口

接口描述

用于下载黑名单数据，该黑名单下发到机具后，在用户手机离线时，机具也可对恶意用户进行拦截。

接口函数

```
public synchronized BlackListResult downloadBlackList(int page_index, int page_size, Date begin_time, Date end_time, String ykt_card_id_flag)
```

输入参数

字段名	类型	说明
page_index	int	分页页码，从 0 开始
page_size	int	一页数据数量
begin_time	date	黑名单生效的开始时间
end_time	date	黑名单生效的结束时间
ykt_card_id_flag	string	补充 ykt_card_id 字段,填空字符即可

输出参数

字段名	类型	说明
blackListResult	BlackListResult	黑名单列表下载结果

类说明

BlackListResult:

字段名	类型	说明
retcode	int	错误码
retmsg	string	错误描述
以下字段在 retcode=0 时返回		
blacklist	ArrayList<BlackItem>	黑名单列表

blacklist:

字段名	类型	说明
open_id	string	用户卡 id
exprie_time	int	失效时间
ykt_card_id	string	商户侧用户卡号(商户分配)

错误码

错误码	错误描述	解决方案
0	成功	
非 0	错误	联系腾讯开发对接人员或查看附录 9.1.2 后台 SDK 错误码(使用腾讯后台 SDK 的商户)

接口示例

```
1.  //**********黑名单接口测试**********
2.  Date begin_time=new Date();
3.  begin_time.setTime(System.currentTimeMillis());
4.  Date end_time=new Date();
5.  end_time.setTime(System.currentTimeMillis() + 24*60*60*1000 );
6.  BlackListResult blackListResult = ccmBackendSDK.downloadBlackList(0,300,begin_time,end_time,""
    );
7.  if(blackListResult.retcode == 0){
8.      for(int i = 0; i < blackListResult.blacklist.size(); i++){
9.          Constant.LogDebug("open_id:" + blackListResult.blacklist.get(i).open_id + " ");
10.         Constant.LogDebug("exprie_time:" + blackListResult.blacklist.get(i).exprie_time + " ");
11.         Constant.LogDebug("ykt_card_id:" + blackListResult.blacklist.get(i).ykt_card_id + "\n")
    ;
12.     }
13.  }else{
14.     Constant.LogDebug("retcode:"+blackListResult.retcode+"\n");
15.     Constant.LogDebug("retmsg:"+blackListResult.retmsg+"\n");
16.  }
```

3.2.5 申请扣款接口

接口描述

该接口用于，用户扫码乘车后，商户向腾讯申请扣款。单次计费场景下，一次扫码产生一笔交易。分段计费场景下，一次乘车有出入站两次扫码。

注：订单申请扣款失败，如需重推请不要更换商户订单号。

接口函数

```
public synchronized OrderPayApplyResult orderPayApply(OrderPayInfo ord
er)
```

输入参数

字段名	类型	说明
order	OrderInfo	订单信息

类说明

OrderInfo:

字段名	类型	长度	必填	说明
ykt_sub_id	string	8-64	O	乘车码子商户号,由腾讯提供(商户可能存在多个运营主体，这里填收款的子商户号)
ykt_sub_sign	string		O	乘车码子商户号签名，商户用 ykt_sub_id 的私钥，对除

				了 sign 以外的全字段计算签名。 如果商户传了，腾讯侧可以验证签名，防止商户传错子商户号，造成资金损失。
ykt_trx_id	string	10-64	M	商户订单号
card_id	string	16	M	用户卡号
total_fee	int	4	M	商户票面金额，单位分 total_fee = collect_fee + discount_fee
collect_fee	int	4	M	商户实际收款金额，单位分
discount_fee	int	4	M	商户减免金额，单位分
ykt_trx_desc	string	1-128	M	商户订单描述： 微信订单展示的订单描述信息
city_id	string	6	M	城市编号
exp_type	int	4	M	异常订单类型 0 - 正常 1 - 单边进站 2 - 单边出站 3 - 单边进站，客服补出站 4 - 单边出站，客服补进站 5 - 分段计费正常订单，行程超时 6 - 超时罚款订单 7 - 进站清本站订单 8 - 补登入站(用户输入进站信息) 9 - 补登出站(用户输入出站信息) 10 -BOM 更新进出站状态 使用场景详见，附录 A[订单类型场景]
charge_type	string	1-64	M	计费类型： 0 - 一次扫码计费 1 - 分段扫码计费
charge_data	ArrayList<ChargeData>		M	计费数据：ChargeData

类说明

乘车计费数据：ChargeData

名称	类型	长度	必填	描述
record_type	int	4	M	1-扫码进站 2-扫码出站 3-BOM 处理/用户补登
record_class	int	4	0	0 - 无扫码数据 1 - 腾讯扫码数据 2 - 保留 3 - 保留 4 - 交通部扫码数据 5 - 保留 6 - 保留

record_time	int	4	M	记录时间，Unix 时间戳
record_data	string	1-1024	M	扫码数据
terminal_id	string	1-64	M	闸机编号
scan_no	string	1-64	M	扫码流水号
station_id	string	1-64	C	车站站点 id,当计费类型为分段计费时必填。
station_name	string	1-128	C	车站站点名,当计费类型为分段计费时必填。
trip_num	string	0-128	O	行程 id
line_no	string	0-128	O	线路号
line_name	string	0-128	O	线路名
vehicle_no	string	0-128	O	车辆编号
plate_no	string	0-128	O	车牌号
driver_id	string	0-128	O	司机编号
longitude	string	0-128	O	经度
latitude	string	0-128	O	维度
qrcode_type	string	1	M	二维码类型，跟 record_class 对应

输出参数

字段名	类型	说明
reportOrderResult	ReportOrderResult	订单上报结果

类说明

ReportOrderResult:

字段名	类型	长度	说明
retcode	int	4	错误码
retmsg	string		错误描述
以下字段在 retcode=0 时返回			
ykt_trx_id	string	10-64	商户订单号
order_id	string	10-64	乘车码订单号
status	int	4	扣款受理结果，详见，附录 A[订单异常拦截] 0 - 成功 3 - 单边账未达推送时间 4 - 交易金额超限 5 - 请求数据错误 10 - 扫码时间超过推送时间范围 11 - 重复扫码行为 12 - 用户欠费超限，禁止付款 99 - 系统错误，请重新申请扣款

错误码

错误码	错误描述	解决方案
0	成功	
非 0	错误	联系腾讯开发对接人员或查看附录 9.1.2 后台 SDK 错误码(使用腾

		讯后台 SDK 的商户)

注：retcode=0 时需要对扣费受理结果状态(status)进行判断，status！=0 时，商户需保存返回数据便于后期问题定位。

接口示例

```
1.  //**********申请扣款接口**********
2.  OrderPayInfo order = new OrderPayInfo();
3.  order.ykt_trx_id = "01234567892037";                    // 商户订单号
4.  order.card_id = "301030C7F720A767";                     // 用户卡号
5.  order.total_fee = 100;                                  // 商户票面金额
6.  order.collect_fee = 100;                                // 商户实际收款金额
7.  order.discount_fee = 0;                                 // 商户减免金额
8.  order.ykt_trx_desc = "describe";                        // 商户订单描述  微信订单展示的订单描述信息
9.  order.exp_type = 0;                                     // 订单异常类型
10. order.charge_type = "1";                                // 计费类型
11. order.city_id = "440100";                               //城市号
12. ChargeData in_charge_data = new ChargeData();           //计费数据
13. in_charge_data.record_type = 1;                         //计费类型
14. in_charge_data.record_class = 1;                        //扫码数据类型
15. in_charge_data.record_time = 1536150668;                //记录时间，Unix 时间戳
16. in_charge_data.record_data = "AAEBAh7lo76tiFsn8IqTHNtFWzC5Vu7Fz6Pz7yekmAMOON83LtcWWQ6dTJBN6pkY
    dLxkBjt/BcRHEdV4gJtLk23QLiR/XvXaMIaSx25MzY+tkRqmGM9YBDErkL/pRoyVr9PcEg==";        //扫码数据
17. in_charge_data.station_id = "10";                       //站点编号
18. in_charge_data.station_name = "1 站";                    //站点名
19. in_charge_data.terminal_id = "789012";                  //机具或闸机 id
20. in_charge_data.scan_no = "12345678";                    //扫码流水号
21. in_charge_data.trip_num = "12345678";                   //行程 id
22. in_charge_data.line_no = "12345678";                    //线路号
23. in_charge_data.line_name = "线路名";                     //线路名
24. in_charge_data.vehicle_no = "12345678";                 //车辆编号
25. in_charge_data.plate_no = "12345678";                   //车牌号
26. in_charge_data.driver_id = "12345678";                  //司机编号
27. in_charge_data.longitude = "12345678";                  //经度
28. in_charge_data.latitude = "12345678";                   //纬度
29. in_charge_data.qrcode_type = "1";
30. order.charge_data.add(in_charge_data);                  //进站计费数据
31. OrderPayApplyResult orderPayApplyResult = ccmBackendSDK.orderPayApply(order);
32. if(orderPayApplyResult.retcode != 0) {                   //retcode!=0 保存日志以便于定位问题
33.     Constant.LogDebug("retcode:"+orderPayApplyResult.retcode+"\n");
34.     Constant.LogDebug("retmsg:"+orderPayApplyResult.retmsg+"\n");
35.     Constant.LogDebug("申请扣款接口失败\n");
36. }else{
37.     Constant.LogDebug("retcode:"+orderPayApplyResult.retcode+"\n");
```

```
38.     Constant.LogDebug("retmsg:"+orderPayApplyResult.retmsg+"\n");
39.     Constant.LogDebug("ykt_trx_id:"+orderPayApplyResult.ykt_trx_id+"\n");
40.     Constant.LogDebug("order_id:"+orderPayApplyResult.order_id+"\n");
41.     Constant.LogDebug("status:"+orderPayApplyResult.status+"\n");
42.     if(orderPayApplyResult.status==0) {
43.         Constant.LogDebug("申请扣款接口成功\n");
44.     }else{
45.         //status! =0 保存日志以便于定位问题
46.         Constant.LogDebug("申请扣款接口失败\n");
47.     }
48. }
```

3.2.6 查询扣款接口

接口描述：

订单查询接口，在商户对订单状态不确认时，可通过该接口查询订单信息。例如：商户申请扣款出现网络抖动、扣费请求未收到应答等异常问题，无法确认订单是否推成功，可通过查单接口查询订单状态。

接口函数

```
public synchronized OrderQueryResult orderPayQuery(String ykt_trx_id,String order_id)
```

输入参数

字段名	类型	长度	必填	说明
ykt_trx_id	string	10-64	C	商户订单号,order_id 为空字符串时,ykt_trx_id 为必传
order_id	string	10-64	C	乘车码订单号，ykt_trx_id 为空字符串时，order_id 为必传

输出参数

字段名	类型	说明
OrderQueryResult	OrderQueryResult	查询扣款结果

类说明 OrderQueryResult:

字段名	类型	说明
retcode	int	错误码
retmsg	string	错误描述
以下字段在 retcode=0 时返回		
order_exist	int	订单是否存在 0 - 不存在 1 - 存在

以下字段在 order_exist=1 时返回		

字段名	类型	说明
ykt_trx_id	string	商户订单号
order_id	string	乘车码订单号
order_time	datetime	请款时间
card_id	string	用户卡号
city_id	string	城市 id
total_fee	int	票面金额，单位分
collect_fee	int	实收金额，单位分
discount_fee	int	减免金额，单位分
exp_type	int	订单异常类型 0 - 正常 1 - 单边进站 2 - 单边出站 3 - 单边进站，客服补出站 4 - 单边出站，客服补进站 5 - 分段计费正常订单，行程超时 6 - 超时罚款订单 7 - 进站清本站订单 8 - 补登入站(用户输入进站信息) 9 - 补登出站(用户输入出站信息) 10 -BOM 更新进出站状态 使用场景详见，附录 A[订单类型场景]
charge_type	int	计费类型： 0 - 一次扫码计费 1 - 分段扫码计费
status	int	扣款受理结果
desc	string	扣款受理描述

错误码

错误码	错误描述	解决方案
0	成功	
非 0	错误	联系腾讯开发对接人员或查看附录 9.1.2 后台 SDK 错误码(使用腾讯后台 SDK 的商户)
	order_exist=0	订单不存在

注：retcode 为 0 时，order_exist 为 0 表示订单不存在。

接口示例

```
//**********查询扣款单接口**********
String ykt_trx_id = "20180913005639"; // 商户订单号
String order_id = "";
OrderQueryResult orderQueryResult = ccmBackendSDK. orderPayQuery(ykt_trx_id,order_id);
if(orderQueryResult.retcode != 0){
    Constant.LogDebug("retcode:"+orderQueryResult.retcode+"\n");
    Constant.LogDebug("retmsg:"+orderQueryResult.retmsg+"\n");
    Constant.LogDebug("查询扣款单接口失败\n");
}else{
    Constant.LogDebug("retcode:"+orderQueryResult.retcode+"\n");
    Constant.LogDebug("retmsg:"+orderQueryResult.retmsg+"\n");
    if(orderQueryResult.order_exist == 0){
        Constant.LogDebug("查询扣款单接口成功,但是订单不存在\n");
    }else{
        Constant.LogDebug("ykt_trx_id:"+orderQueryResult.ykt_trx_id+"\n");
        Constant.LogDebug("order_id:"+orderQueryResult.order_id+"\n");
        Constant.LogDebug("order_time:"+orderQueryResult.order_time+"\n");
        Constant.LogDebug("card_id:"+orderQueryResult.card_id+"\n");
        Constant.LogDebug("city_id:"+orderQueryResult.city_id+"\n");
        Constant.LogDebug("total_fee:"+orderQueryResult.total_fee+"\n");
        Constant.LogDebug("collect_fee:"+orderQueryResult.collect_fee+"\n");
        Constant.LogDebug("discount_fee:"+orderQueryResult.discount_fee+"\n");
        Constant.LogDebug("exp_type:"+orderQueryResult.exp_type+"\n");
        Constant.LogDebug("charge_type:"+orderQueryResult.charge_type+"\n");
        Constant.LogDebug("status:"+orderQueryResult.status+"\n");
    }
    Constant.LogDebug("查询扣款单接口成功\n");
}
```

3.2.7 申请退款接口

接口说明：

当交易发生之后一段时间内，由于乘客或者商户的原因需要退款时，商户可以通过退款接口将支付款退还给乘客，乘车码将在收到退款请求并且验证成功之后，将支付款按原路退到乘客账号上。

注意：
1. 交易时间超过一年的订单无法提交退款
2. 允许退款金额小于或等于商户实际需要收款的金额
3. 每笔订单只允许退款一次

接口函数：

```
public synchronized OrderRefundApplyResult orderRefundApply(OrderPayInfo order)
```

输入参数

字段名	类型	说明
order	OrderPayInfo	订单信息

类说明

OrderInfo:

字段名	类型	长度	必填	说明
ykt_refund_id	string	10-64	O	商户侧生成的退款单号
ykt_trx_id	string	10-64	M	商户订单号
card_id	string	16	M	用户卡号
total_fee	int	4	M	申请扣款时的票面金额，单位分
refund_fee	int	4	M	退款金额，单位分
refund_desc	string	1-128	O	微信退款订单展示的描述信息

输出参数

字段名	类型	说明
orderRefundApplyResult	OrderRefundApplyResult	订单上报结果

类说明

OrderRefundApplyResult:

字段名	类型	长度	说明
retcode	int	4	错误码
retmsg	string		错误描述

错误码

错误码	错误描述	解决方案
0	成功	
非 0	错误	联系腾讯开发对接人员或查看附录 9.1.2 后台 SDK 错误码(使用腾讯后台 SDK 的商户)

接口示例

```
1.  //**********申请退款接口**********
2.  OrderPayInfo order2 = new OrderPayInfo();
3.  order2.ykt_refund_id = "0123456789";                    // 商户退款订单号，选填
4.  order2.ykt_trx_id = "11553568106498265030122020190326";  // 商户订单号
5.  order2.card_id = "2007CFC75CE90811";                    // 用户卡 id
6.  order2.total_fee = 200;
7.  order2.refund_fee = 180;
8.  order2.refund_desc = "test";
9.  OrderRefundApplyResult orderRefundApplyResult = ccmBackendSDK.orderRefundApply(order2);
10. if(orderRefundApplyResult.retcode != 0){
11.     Constant.LogDebug("retcode:"+orderRefundApplyResult.retcode+"\n");
12.     Constant.LogDebug("retmsg:"+orderRefundApplyResult.retmsg+"\n");
13.     Constant.LogDebug("退款接口失败\n");
```

```
14. }else{
15.     Constant.LogDebug("退款接口成功\n");
16. }
```

3.2.8 查询退款接口

接口说明：

订单查询接口，在商户无法确认退款操作是否成功时，可通过该接口查询退款状态及订单信息。

接口函数

```
public synchronized OrderRefundQueryResult orderRefundQueryResult(String ykt_trx_id,String ykt_refund_id)
```

输入参数

字段名	类型	长度	必填	说明
ykt_trx_id	string	10-64	M	商户订单号
ykt_refund_id	string	10-64	O	商户退款单号

输出参数

字段名	类型	说明
orderRefundQueryResult	OrderRefundQueryResult	查询扣款结果

类说明 OrderRefundQueryResult：

字段名	类型	说明
retcode	int	错误码
retmsg	string	错误描述
以下字段在 retcode=0 时返回		
order_exist	int	订单是否存在
以下字段在 order_exist=1 时返回		
ykt_trx_id	string	商户订单号
total_fee	int	票面金额，单位分
collect_fee	int	实收金额，单位分
refund_fee	int	退款金额，单位分
refund_status	int	退款状态： 40 - 退款中 50 - 退款成功 51 - 退款失败
err_desc	string	退款失败时存在，退款失败描述

错误码

错误码	错误描述	解决方案
0	成功	

非 0	错误	联系腾讯开发对接人员

注：retcode 为 0 时，order_exist 为 0 表示订单不存在。

接口示例

```
1.  //**********************查询退款单接口**********************
2.  String ykt_refund_id = "20190324221010123";                  //商户退款订单号，可选
3.  String ykt_trx_id = "11553568106498265030122020190326";     // 商户订单号
4.  OrderRefundQueryResult orderRefundQueryResult = ccmBackendSDK.orderRefundQueryResult(ykt_trx_i
    d, ykt_refund_id);
5.  if(orderRefundQueryResult.retcode != 0){
6.     Constant.LogDebug("retcode:"+orderRefundQueryResult.retcode+"\n");
7.     Constant.LogDebug("retmsg:"+orderRefundQueryResult.retmsg+"\n");
8.     Constant.LogDebug("查询退款接口失败\n");
9.  }else{
10.     Constant.LogDebug("retcode:"+orderRefundQueryResult.retcode+"\n");
11.     Constant.LogDebug("retmsg:"+orderRefundQueryResult.retmsg+"\n");
12.     if(orderRefundQueryResult.order_exist == 0){
13.         Constant.LogDebug("查询退款接口成功,但是订单不存在\n");
14.     }else{
15.         Constant.LogDebug("ykt_trx_id:"+orderRefundQueryResult.ykt_trx_id+"\n");
16.         Constant.LogDebug("total_fee:"+orderRefundQueryResult.total_fee+"\n");
17.         Constant.LogDebug("collect_fee:"+orderRefundQueryResult.collect_fee+"\n");
18.         Constant.LogDebug("refund_fee:"+orderRefundQueryResult.refund_fee+"\n");
19.         Constant.LogDebug("refund_status:"+orderRefundQueryResult.refund_status+"\n");
20.         Constant.LogDebug("err_desc:"+orderRefundQueryResult.err_desc+"\n");
21.     }
22.     Constant.LogDebug("查询退款接口成功\n");
23. }
```

3.2.9 下载对账单接口

接口说明：

通过该接口下载商户历史扣款请求清单，对账单账单用于核对腾讯后台收到的与商户后台申请扣款订单数量、金额。例如掉单、网络错误等导致商户侧和腾讯后台收到的订单数目不一致，通过对账单核对后可补齐。

注意：1. 腾讯后台未收到请求的扣款单不会出现在对账单中。

2. 腾讯后台在 10 点完成前一天的对账单生成，建议商户 10 点后再下载获取。

3. 对账单中涉及的金额单位为“元”。

4. 商户后台只能下载最近 31 日内的账单。

5. 该接口主要用于商户订单的对账，确保腾讯后台收到与商户后台申请扣款的订单数量及金额的一致性。该对账单不用于财务的资金流对账，财务资金流对账请下载资金账单。

接口函数

```
public synchronized OrderBillsResult downloadOrderBills(String
```

version,Date date,String download_filepath)

输入参数

字段名	类型	长度	必填	说明
version	string		M	版本号，支持 3.0
date	date		M	账单日期
download_filepath	string		M	文件路径

返回参数

Content-Type=application/octet-stream 数据将以字节流的形式下载到本地，内容为 utf-8 纯文本，LF 换行。

账单文件内容示例：

商户订单号	乘车码订单号	接收请款时间	票面金额(元)	实收金额(元)	减免金额(元)	赔付金额(元)	扣款受理结果	扣款受理描述
1000000001	2000000001	2018-08-01 01:11:59	2.00	1.80	0.20	0.00	0	OK
1000000002	2000000002	2018-08-01 01:11:59	2.00	1.80	0.20	0.00	0	OK
1000000003	2000000003	2018-08-01 01:11:59	2.00	1.80	0.20	0.00	0	OK
1000000004	2000000004	2018-08-01 01:11:59	2.00	1.80	0.20	0.00	0	OK
总条数	票面金额(元)	实收金额(元)	减免金额(元)	赔付金额(元)				
4	8.00	7.20	0.80	0. 00				

扣款受理结果，请参考【订单拦截规则】

Content-Type 不为 application/octet-stream 时，就以 json 格式返回错误信息

字段名	类型	长度	说明
retcode	int	4	错误码
retmsg	string		错误描述

错误码

错误码	错误描述	解决方案
0	成功	
非 0	错误	联系腾讯开发对接人员或查看附录 9.1.2 后台 SDK 错误码(使用腾讯后台 SDK 的商户)

接口示例：

```
1.  /*-----------------下载对账单接口-----------------------*/
2.  Date date1=new Date();
3.  date1.setTime(System.currentTimeMillis() - 24*60*60*1000 );
4.  SimpleDateFormat simpleDateFormat1 = new SimpleDateFormat("yyyyMMdd");
5.  String version="3.0";
6.  OrderBillsResult orderBillsResult = ccmBackendSDK.downloadOrderBills(version,date1,"D:\\worksp
      ace\\orderBillsFile_0911.csv");
```

```
7.  if(orderBillsResult.retcode != 0){
8.      Constant.LogDebug("retcode:"+orderBillsResult.retcode);
9.      Constant.LogDebug("retmsg:"+orderBillsResult.retmsg);
10.     Constant.LogDebug("下载对账单接口失败");
11.  }
```

3.2.10 下载资金账单接口

接口说明：

财务资金账单接口，可以下载自 2018 年 1 月 1 日起的账单。

注：1. 资金账单中的数据反映的是商户资金变动情况。

2. 资金账单中涉及的金额字段单位为“元”。

3. 腾讯后台在 10 点完成前一天的对账单生成，建议商户 10 点后再下载获取。

接口函数

```
public synchronized FundBillsResult downloadFundBills(String version,Date date,String download_filepath)
```

输入参数

字段名	类型	长度	必填	说明
version	string		M	版本号，支持 3.0、4.0
date	date		M	账单日期
download_filepath	string		M	文件路径

输出参数

Content-Type=application/octet-stream 数据将以字节流的形式下载到本地，内容为 utf-8 纯文本，LF 换行。

注意：一次扣款可能对应多条资金流水 。

version=3.0，账单文件内容示例：

商户订单号	乘车码订单号	商户平台交易单号	商户号	订单金额(元)	退款金额(元)	交易时间	手续费(元)	订单类型
1000000001	2000000001	3000000001	123456	1.80	0.00	*	0	用户应付
1000000002	2000000002	3000000002	123456	2.00	0.00	*	0	用户应付
1000000003	2000000003	3000000003	123456	2.00	0.00	*	0.01	用户应付
1000000004	2000000004	3000000004	654321	2.00	0.00	*	0.01	车票
1000000005	2000000005	3000000005	654321	0.00	2.00	*	-0.01	车票
商户号	交易单数合计	交易额合计(元)	退款合计(元)	手续费合计(元)				
123456	3	5.80	0.00	0.01				
654321	2	2.00	2.00	0.00				
总交易单数	总交易额(元)	总退款额(元)	总手续费(元)					
5	7.80	2.00	0.01					

version=3.0 资金账单文件说明：

字段	说明
商户订单号	申请扣款时传入的商户订单号 ykt_trx_id
乘车码订单号	申请扣款时腾讯返回的乘车码订单号 order_id
商户平台交易单号	腾讯在微信、财付通进行资金操作的交易单号
商户号	商户在微信、财付通开通的商户号
订单金额	支付的金额。表示支付流水，此时退款金额为 0.00
退款金额	退款的金额。表示退款流水，此时订单金额为 0.00
交易时间	乘车码在微信、财付通成功扣款或退款的时间。 与微信或财付通商户平台下载的账单中的时间一致。
手续费	当订单金额>0 时，为正值，表示订单中含的手续费。 当退款金额>0 时，为负值，表示退款中含的手续费。
订单类型	一个商户号可能因为车票等优惠活动被拆成多笔 如：1 笔商户订单，对应 1 笔用户应付单 + 1 笔车票单 用户应付：用户应该付款的金额，可能为腾讯垫资，请用商户号区分 车票：由腾讯车票支付的金额部分
	第二类为商户号维度的统计信息
商户号	商户在微信、财付通开通的商户号
交易单数合计	商户号交易总笔数
交易额合计	商户号总订单金额
退款合计	商户号总退款金额
手续费合计	商户号总手续费
	第三类为当日汇总统计
总交易单数	当日总交易笔数
总交易额	当日总交易额
总退款额	当日总退款额
总手续费	当日总手续费

version=4.0，账单文件内容示例：

商户订单号	乘车码订单号	商户平台订单号	商户平台交易单号	商户退款订单号	商户平台退款订单号	商户平台退款交易单号	商户号	乘车码卡号
app_id	open_id	订单金额(元)	商户平台优惠金额(元)	退款金额(元)	商户平台优惠退款金额(元)	交易时间	手续费(元)	费率
订单类型	付款银行	货币种类	商户订单描述	商户自定义数据				
1000000001	2000000001	3000000001	4000000001	*	*	*	123456	30202F3813109256
wx2421b1c4370ec43b	085e9858e3ba5186aafcbaed1	1.80	0.20	0.00	0.00	2018-8-11:11:59	0.00	0.50%
用户应付	CFT	CNY	腾讯乘车码	*				
1000000002	2000000002	3000000002	4000000002	*	*	*	1234	30202F

							56	3813109256
wx2421b1c4370ec43b	085e9858e3ba5186aafcbaed1	1.80	0.20	0.00	0.00	2018-8-11:11:59	0.00	0.50%
用户应付	CFT	CNY	腾讯乘车码	*				
1000000003	2000000003	3000000003	4000000003	*	*	*	654321	30202F3813109256
wx2421b1c4370ec43b	085e9858e3ba5186aafcbaed1	1.80	0.20	0.00	0.00	2018-8-11:11:59	0.00	0.50%
用户应付	CFT	CNY	腾讯乘车码	*				
						*		
商户号	交易单数合计	交易额合计(元)	退款合计(元)	手续费合计(元)				
123456	2	3.60	0.00	0.01				
654321	1	1.80	0.00	0.00				
总交易单数	总交易额(元)	总退款额(元)	总手续费(元)					
3	5.40	0.00	0.01					

version=4.0 资金账单文件说明：

字段	说明
商户订单号	申请扣款时传入的商户订单号 ykt_trx_id
乘车码订单号	申请扣款时腾讯返回的乘车码订单号 order_id
商户平台订单号	乘车码在微信、财付通进行扣款操作的商户订单号
商户平台交易单号	乘车码在微信、财付通进行扣款操作的交易单号
商户退款订单号	申请退款时传入的 ykt_refund_id，没传则默认为 ykt_trx_id
商户平台退款订单号	乘车码在微信进行退款操作的退款订单号，财付通渠道为空
商户平台退款交易单号	乘车码在微信、财付通进行退款操作时的交易单号
商户号	商户在微信、财付通开通的商户号
乘车码卡号	申请扣款时传入的 card_id
app_id	支付渠道 app_id
open_id	支付渠道 open_id，财付通渠道为空
订单金额	支付的金额。表示支付流水，此时退款金额为 0.00
商户平台优惠金额	乘车码在代扣时给与用户的优惠金额
退款金额	退款的金额。表示退款流水，此时订单金额为 0.00
商户平台优	乘车码在退款时，属于优惠金额的退款部分

惠退款金额	
交易时间	乘车码在微信、财付通成功扣款或退款的时间。 与微信或财付通商户平台下载的账单中的时间一致。
手续费	当订单金额>0 时，为正值，表示订单中含的手续费。 当退款金额>0 时，为负值，表示退款中含的手续费。
费率	手续费费率
订单类型	一个商户号可能因为车票等优惠活动被拆成多笔 如：1 笔商户订单，对应 1 笔用户应付单 +1 笔车票单 用户应付：用户应该付款的金额，可能为腾讯垫资，请用商户号区分 用户退款：用户应付对应的退款 车票：由腾讯车票支付的金额部分 车票退款：车票对应的退款
付款银行	参考附录：银行列表
货币种类	CNY：人民币
商户订单描述	申请扣款的 ykt_trx_desc
商户自定义数据	保留，未使用
	第二类为商户号维度的统计信息
商户号	商户在微信、财付通开通的商户号
交易单数合计	商户号交易总笔数
交易额合计	商户号总订单金额
退款合计	商户号总退款金额
手续费合计	商户号总手续费
	第三类为当日汇总统计
总交易单数	当日总交易笔数
总交易额	当日总交易额
总退款额	当日总退款额
总手续费	当日总手续费

Content-Type 不为 application/octet-stream 时，就以 json 格式返回错误信息

字段名	类型	长度	说明
retcode	int	4	错误码
retmsg	string		错误描述

错误码

错误码	错误描述	解决方案
0	成功	
非 0	错误	联系腾讯开发对接人员或查看附录 9.1.2 后台 SDK 错误码(使用腾讯后台 SDK 的商户)

接口示例：

```
1.  /*--------------下载资金账单接口--------------------*/
2.  Date date2=new Date();
3.  date2.setTime(System.currentTimeMillis() - 24*60*60*1000 );
4.  String version="3.0";  //或 4.0
```

```
5.  FundBillsResult fundBillsResult = ccmBackendSDK.downloadFundBills(version,date2,"D:\\workspace
    \\fundBillsFile_0912.csv");
6.  if(fundBillsResult.retcode != 0){
7.      Constant.LogDebug("retcode:"+fundBillsResult.retcode);
8.      Constant.LogDebug("retmsg:"+fundBillsResult.retmsg);
9.      Constant.LogDebug("下载资金账单接口失败");
10. }
```

3.2.11 推送行程数据接口(可选)

接口说明：

推送行程相关的数据到腾讯后台，该接口仅适用于分段计费场景，商户可以选择是否接入该接口。

接口函数

```
public synchronized TripInfoPushResult pushTripInfo(metro_trip_info);
```

输入参数

字段名	类型	说明
metro_trip_info	MetroTripInfo	推送信息

类说明

MetroTripInfo:

字段名	类型	长度	必填	说明
card_id	string	16	是	账户 id(腾讯分配)
ykt_card_id	string	16-64	是	卡 id(商户分配)
action_id	string	1-128	是	行程 id，用于去重
action_src	string	1-128	是	进/出站记录类型 1：扫码进/出站 2：客服补进/出站
ticket_seqno	string	1-64	可选	行程数据 id
line_id	string	1-64	可选	出入站的线路 id
line_name	string	1-64	可选	出入站的线路名
station_id	string	1-64	是	出入站的站点编号
station_name	string	1-64	是	出入站的站点名称
device_id	string	1-64	可选	设备 id
scan_time	string	1-128	是	扫码时间，格式：yyyy-MM-dd HH:mm:ss
scan_type	string	1-2	是	扫码类型
biz_amount	string	0-4	可选	交易金额，单位分
biz_subject	string	0-512	可选	格式：线路-车站-设备
biz_time	string	1-128	可选	格式：yyyy-MM-dd HH:mm:ss
record	string	0-1024	可选	扫码记录
record_report_time	string	0-128	是	上报时间

输出参数

字段名	类型	说明
tripInfoPushResult	TripInfoPushResult	订单上报结果

类说明

TripInfoPushResult:

字段名	类型	长度	说明
retcode	int	4	错误码
retmsg	string		错误描述

错误码

错误码	错误描述	解决方案
0	成功	
非 0	错误	联系腾讯开发对接人员或查看附录 9.1.2 后台 SDK 错误码(使用腾讯后台 SDK 的商户)

SDK 请求示例

```
1.  //**********************推送行程数据测试**********************
2.  MetroTripInfo metro_trip_info = new MetroTripInfo();
3.  metro_trip_info.card_id = "3010F8E28C7A8772";        //账户 id(腾讯分配)
4.  metro_trip_info.ykt_card_id = "3100910178103216";   //卡 id(商户分配)
5.  metro_trip_info.action_id = "1234567890";           //行程 id，用于去重
6.  metro_trip_info.action_src = "1";                   //进/出站记录类型，1：扫码进/出站 2:客服补进/出站
7.  metro_trip_info.ticket_seqno = "28766577";          //行程数据 id
8.  metro_trip_info.line_id  = "2";                     //上次出入站的线路 id
9.  metro_trip_info.line_name = "地铁线 ssss";          //上次出入站的线路名
10. metro_trip_info.station_id = "123456";              //上次出入站的站点编号
11. metro_trip_info.station_name = "Station A";         //上次出入站的站点名称
12. metro_trip_info.device_id = "1234567890";           //设备 id
13. metro_trip_info.scan_time = "2018-08-08 02:23:34"; //扫码时间
14. metro_trip_info.scan_type = "1";                    //扫码类型
15. metro_trip_info.biz_amount= "200";                  //交易金额
16. metro_trip_info.biz_subject = "1 路-001 设备";      //格式：线路-车站-设备
17. SimpleDateFormat simpleDateFormat1 = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
18. Date date1 = new Date();
19. date1.setTime(System.currentTimeMillis());
20. metro_trip_info.biz_time = simpleDateFormat1.format(date1);  //交易时间
21. metro_trip_info.record_report_time = "2018-07-27 17:00:34";  //上报时间
22. TripInfoPushResult tripInfoPushResult = ccmBackendSDK.pushTripInfo(metro_trip_info);   //调用推送行程数据接口
23. if(tripInfoPushResult.retcode != 0){
```

```
24.     Constant.LogDebug("retcode:"+tripInfoPushResult.retcode+"\n");  //返回码
25.     Constant.LogDebug("retmsg:"+tripInfoPushResult.retmsg+"\n");    //返回消息
26.     Constant.LogDebug("推送行程接口失败\n");
27. }else{
28.     Constant.LogDebug("retcode:"+tripInfoPushResult.retcode+"\n");
29.     Constant.LogDebug("retmsg:"+tripInfoPushResult.retmsg+"\n");
30.     Constant.LogDebug("推送行程接口成功\n");
31. }
```

3.2.12 单边补登消息通知接口(可选)

接口说明：

推送行程相关的数据到腾讯后台，该接口仅适用于分段计费场景，商户可以选择是否接入该接口。

接口函数

```
public synchronized TripInfoPushResult regNotify (regs);
```

输入参数

字段名	类型	说明
regs	ArrayList<RegInfo>	补登信息

类说明

RegInfo:

字段名	类型	长度	必填	说明
ykt_reg_id	string	32-64	是	合作伙伴补登流水号
action_id	string	1-128	是	行程 id
card_id	string	16	是	账户 id
city_code	string	1-8	是	城市编码(腾讯提供，国标城市代码)
line_id	string	1-64	是	线路 id
line_type	string	1-64	是	线路类型
line_name	string	1-64	是	线路名称
station_id	string	1-64	是	站点名称
station_name	string	1-64	是	站点 id
device_id	string	8-64	可选	设备 id
scan_time	datetime	1-128	是	扫码时间,Unixt 时间戳
max_fee	int	4	是	最大金额，单位分
reg_type	int	4	是	补登类型
expire_time	datetime	1-128	是	补登超时时间

输出参数

字段名	类型	说明
regNotifyResult	RegNotifyResult	补登结果

类说明

RegNotifyResult:

字段名	类型	说明
retcode	int	错误码
retmsg	string	错误描述
以下字段在 retcode=0 时返回		
result	string	返回结果
message	string	返回结果消息
reg_id	string	腾讯补登订单号
ykt_reg_id	string	商户补登订单号

错误码

错误码	错误描述	解决方案
0	成功	
非 0	错误	联系腾讯开发对接人员或查看附录 9.1.2 后台 SDK 错误码(使用腾讯后台 SDK 的商户)

接口示例

```
1.  //**********************补登消息接口**********************
2.  ArrayList<RegInfo> regs = new ArrayList<RegInfo>();
3.  RegInfo regInfo = new RegInfo();
4.  regInfo.ykt_reg_id = "20001400200000201808012050000025";    //合作伙伴补登流水号
5.  regInfo.action_id = "1234567890";                //行程 id
6.  regInfo.card_id = "2400F1F491457510";            //账户 id
7.  regInfo.city_code = "330100";                    //城市编码
8.  regInfo.line_id = "123";                         //线路 id
9.  regInfo.line_type = "2";                         //线路类型
10. regInfo.line_name = "commonline";                //线路名称
11. regInfo.station_name = "city_name";              //站点名称
12. regInfo.station_id = "5";                        //站点 id
13. regInfo.device_id = "123456";                    //机具 id
14. regInfo.scan_time = 1533563465;                  //扫码时间
15. regInfo.max_fee = 100;                           //最大金额
16. regInfo.reg_type = 2;                            //补登类型
17. regInfo.expire_time = 1533573465;                //补登超时时间
18. regs.add(regInfo);
19. RegNotifyResult regNotifyResult = ccmBackendSDK.regNotify(regs);        //调用补登接口
20. if(regNotifyResult.retcode != 0) {
21.     Constant.LogDebug("retcode:" + regNotifyResult.retcode+"\n");
22.     Constant.LogDebug("retmsg:" + regNotifyResult.retmsg+"\n");
23.     Constant.LogDebug("result:" + regNotifyResult.result+"\n");          //返回结果
24.     Constant.LogDebug("message:" + regNotifyResult.message+"\n");        //返回结果消息
25.     Constant.LogDebug("reg_id:" + regNotifyResult.reg_id+"\n");          //腾讯补登订单号
```

```
26.     Constant.LogDebug("ykt_reg_id:" + regNotifyResult.ykt_reg_id+"\n"); //合作伙伴补登订单号
27.     Constant.LogDebug("补登消息接口失败\n");
28.   }else{
29.     Constant.LogDebug("retcode:" + regNotifyResult.retcode+"\n"); //返回码
30.     Constant.LogDebug("retmsg:" + regNotifyResult.retmsg+"\n");   //返回消息
31.     Constant.LogDebug("result:" + regNotifyResult.result+"\n");   //返回结果
32.     Constant.LogDebug("message:" + regNotifyResult.message+"\n"); //返回结果消息
33.     Constant.LogDebug("reg_id:" + regNotifyResult.reg_id+"\n");   //腾讯补登订单号
34.     Constant.LogDebug("ykt_reg_id:" + regNotifyResult.ykt_reg_id+"\n"); //合作伙伴补登订单号
35.      Constant.LogDebug("补登消息接口成功\n");
36.   }
```

3.2.13 手机号查询卡 id 接口

接口描述：

商户可以通过用户手机号，查询用户对应的卡 id。

接口函数

```
public synchronized QueryResult cardIDQuery (phone_encrypt);
```

输入参数

字段	类型	说明
phone_encrypt	string	手机号

输出参数

字段名	类型	说明
queryResult	QueryResult	查询结果

类说明
QueryResult:

字段	类型	说明
retcode	string	返回码 0：成功 非 0：系统错误
retmsg	string	返回消息
cardid	string	用户卡 id
sign	string	平台私钥签名 sha1withrsa

错误码

错误码	错误描述	解决方案
0	成功	
非 0	错误	联系腾讯开发对接人员或查看附

		录 9.1.2 后台 SDK 错误码(使用腾讯后台 SDK 的商户)

接口示例

```
1. /*----------手机号查询卡 id--线上环境接口------*/
2. QueryResult queryResult = ccmBackendSDK.cardIDQuery("13686816615");
3. if (queryResult.retcode == 0) {
4.     Constant.LogDebug("queryResult.cardid" + queryResult.cardid);
5. }else{
6.     Constant.LogDebug("retcode:" + queryResult.retcode + "\n");
7.     Constant.LogDebug("retmsg:" + queryResult.retmsg + "\n");
8.     Constant.LogDebug("手机号查询卡 ID 失败\n");
9. }
```

第4章 沙箱联调环境

4.1 说明

在接入线上正式环境之前，合作伙伴需要在腾讯提供的沙箱环境调通扫码、订单上传、黑名单下载、公钥下载、对账单下载等开放平台接口功能。

4.2 接入方法

1. 沙箱环境调试，使用腾讯提供的乘车码模拟生成器 APP 生成二维码，请联系腾讯技术人员提供；
2. 沙箱环境部署在 220.249.243.193 上，合作伙伴通过配置如下 host 访问沙箱环境进行调试：220.249.243.193 open-wlx.tenpay.com。否则无法调试成功；
3. 正式上线前，调试阶段必须设置 host(220.249.243.193)在沙箱调试；禁止不设置 host 直接在线上调试；
4. 沙箱环境只做接口联调，并不产生扣费；
5. 联调数据：ykt_id:20000007，city_code: 440100；
6. 沙箱调试使用的公私钥全部由腾讯提供，仅作为沙箱调试使用，请联系腾讯技术人员提供。
7. 测试/正式环境开关
   DEBUG_TAG：0 是正式，1 是测试；
   HTTP/HTTPS 开关 SECURE_TAG：0 是 http，1 是 https。

第5章 商户接入要求

5.1 机具功能

1. 机具实现此文档所有机具 SDK 接口对接；
2. 机具实现公钥、MAC 密钥的定时更新机制；
3. 单个二维码不能重复使用，建议通过保留一定时间（大于二维码有效时长即可，建议默认使用 5 分钟）内的二维码数据的摘要，用于防重。注意：机具 SDK 输出的扫码记录不可作为单码去重条件；
4. 机具需要至少每日做一次互联网标准时间校准，商户需要承诺机具时间最大误差；
5. 用户乘车记录应在 5 秒内推单到腾讯商户平台；
6. 机具界面交互建议按照《机具扫码语音及文案建议》实现（见附件 9.2）；
7. 需要对机具有的监控机制，机具无法工作或一直联网失败时，需要及时发现并下线；
8. 需要实现 SDK 后台在线更新的能力。

5.2 机具硬件要求

1. 主控芯片，推荐 Cortex M4-168MHz 以上的主控芯片，保证验码耗时在 300ms 以内。内存需至少预留 5KB 供机具 SDK 使用；
2. 存储空间，机具需提供保存 10W 条黑名单记录(2MB)，和保存 1W 条交易记录(2MB)的存储空间，共计 4MB。实际存储空间可根据具体情况调整。

5.3 服务器要求

服务器应与网络时钟源对齐，服务器必须每天做一次 NTP 时钟校准，保证与互联网时间最大误差小于±10 秒。推荐使用以下国内常用网络时钟源，请勿使用系统默认时钟源。

1.cn.pool.ntp.org
2.cn.pool.ntp.org
3.cn.pool.ntp.org
0.cn.pool.ntp.org
cn.pool.ntp.org
tw.pool.ntp.org

5.4 运维注意事项

机具及后台服务有任何变动或需要升级，需提前知会腾讯，以共同评估影响；有损升级或变动必须在夜间或者用户使用量最少的时候进行，以免引起用户投诉。

第6章 商户后台能力要求

6.1 后台能力要求

公司名称：		填表日期：	
模块	**能力描述**	**分值**	**优先级**
服务容灾（40 分）	应用服务多机容灾能力：当一台应用服务停止服务时，至少可以自动切换到另一台应用服务器，以维持业务正常运转	12	高
	DB 多机容灾能力：当一台 DB 停止服务时，至少可以自动切换到另一台数据完整的 DB，以维持业务正常运转	16	高
	多机房容灾能力：DB 和服务器面署在不同的机房中，当一个机房停止服务时，至少有另一外机房可以维持业务正常运转	2	中
	应用服务与 DB 隔离：应用服务与 DB 分开布署在不同的物理机器上	8	中
	服务隔离：机具订单收取服务、向腾讯后台推单及腾讯接品服务隔离，当任何一个服务停止工作时，其它服务不受影响	2	中
网络容灾（20 分）	商户后台与腾讯后台至少具有两条网络链路，当一条链路不通时，可以自动切换到另一条链路维持业务正常运转，包括网络设备、负载均衡器、应用接入层等不存在单点故障	20	高
应急补推单脚本（10 分）	当向腾讯后台推单服务停止服务时，应具有通过应急补推单脚本对未推送给腾讯后台的记录进行补推的能力，此脚本独立于向腾讯后台推单服务	10	高
监控能力（30 分）	后台订单实时监控能力：推单失败率、失败次数、推单及时性	9	高
	接口错误实时监控能力	9	高
	对服务器磁盘空间使用情况具有实时监控、告警和自动清理能力	3	中
	机具状态实时监控能力：推单情况	4.5	中
	机具扫码错误时，错误码上传后台与统计能力	4.5	中
商户隔离（10 分）	不同业主的后台系统分开布署，当一个商户后台系统停止服务时，不影响其它商户业务的正常运转	10	中

注意：若合作伙伴容灾能力无法满足上述要求，请尽早提出，与腾讯联和商议讨论后按计划进行优化和加强或调整。包括并不限于，db 部署、配置、系统性能、服务可靠性、机房安全性、容灾能力等，以免之后出问题时过于被动并导致负面影响。

6.2 后台能力分值统计

商户需要按照后台能力统计表来计算后台能力，并且**必须**将后台能力分值统计反馈给腾讯。

乘车码商户后台能力分值统计															
公司名称	1	2	3	4	5	6	7	8	9	10	11	12	总分	商户隔离	统计日期

第7章 正式上线流程

7.1 上线流程介绍

7.1.1 腾讯后台发布与线上环境测试

沙箱环境联调完成后，将在线上环境进行测试。
第一步：腾讯后台提供正式通信公钥、APPID；商户后台提供正式通信公钥；商户提供微信商户号和财付通商户号的密钥及证书、退款操作员帐号及密码。
第二步：商户后台进行线上部署。
第三步：腾讯后台发布。
第四步：腾讯后台发布后可进行线上测试。
第五步：测试完成后，商户或机具方邮寄机具至腾讯，寄出前应确认机具是否已经配置好，是否可以在腾讯测试刷码。

7.1.2 产品体验与功能完整性检查

第一步：腾讯配置白名单。
第二步：各方进行机具功能体验，体验用例可参照《机具体验验收表》。
第三步：功能完整性检查。将《机具扫码语音及文案》、《腾讯乘车码机具功能检查表》、《腾讯乘车码后台检查表》、《腾讯乘车码商户后台能力排查表》、《机具体验验收表》发至腾讯。

7.1.3 开放入口

完成产品体验、功能完整性检查后，可放开产品前端入口。
入口放开后，所有用户均可开通使用乘车码。

7.2 通信密钥生成说明

1. 双方通信需互相验证对方的密钥签名，因此上线前需生成并交换双方 RSA 公钥；
2. 腾讯侧公钥命名规则：tx_ccm_*xxx*_public_key.pem 文件；
3. 商户侧公钥建议命名方式：mch_ccm_*xxx*_public_key.pem 公钥文件，xxx 为城市名称；
4. 目前 RSA 算法支持 1024 位，暂不支持 2048；
5. RSA 公私钥请使用 PEM 文件提供；

公私钥生成请使用 openssl 标准命令，在 linux 环境下生成（请勿使用非公开工具生成）
第一步：生成私钥命令：openssl genrsa -out xxx_ccm_private_key.pem 1024
第二步：用私钥生成公钥：openssl rsa -in xxx_ccm_private_key.pem -pubout -out mch_ccm_*xxx*_public_key.pem
下图为示例：

私钥内容示例：

```
-----BEGIN RSA PRIVATE KEY-----
MIICXAIBAAKBgQDGjENfZzAK3yVGrPrQ8Qq3ZEbTs6sOb2qOZHukWMj8EnHifveO
7UAF+I133bcHrpscbCuO9vXngUtvVoIv1uZCWZLS6O/nOpxZjTxKUgJdWHFGLKdI
5MKqNjgYl6fyIkOHjeqOXfjrFIOeZIlC7HIoCObuU3GGJd4OzJjBCIw9BwIDAQAB
AoGAIMCkoIlKIJD8+jKwOWTh6qzNDkE2N5RjmsxOpQDMbuCwDrf5LsxwYkvFMn/N
oKPKDJ8/eoO+vujSQ9SWdOPRCdhyg2ZE/OH3gGoROI3KO6AavgVwqmse6NHQsG5K
hKOTavIoRj17p3WU7gyMxP/31v0889X5+FPn8OVIVIKfOTkCQQDulsJsK4Qs9IgA
yQ2ruYoXj/iCYeWEF1+ZiOqbzd5c+h5JU7Oo3r7YEJYWxR88qQp3+n/u9PogazIX
thk9v/u7AkEA1Ql4gKXy+o1Dw/a3BnxRg74Wv3Kwyo5zfVIKv8EYz+rkxFwUG1Ra
2xVGRhMZCXzYNO4EGLdoPo7cyk9fkhFhJQJALJe+5OrJPIHDvanWRUdbWQYOo4fa
lvKrao3dk5tJuUUCTiA3zxM6xjVrZ/wV14ecrkoCiU3+RfrjFO1zEEiGqwJBALwc
ccA/SK8+v8CxAGoEqkHHPvTsA/nIEWhuYjlx+OPUqQ5ONB7xvIxJW9USlgAPsigb
hniAstbSdVla/wvOttUCQANMxyzmEoGkmNQ2nZNCRIZuBavOoczo4VlvEV6OOnsP
H4C8mbmgB7GOWvtfkM9vcxysxe6+VoiMD39XJeTdwOo=
-----END RSA PRIVATE KEY-----
```

公钥内容示例：

```
-----BEGIN PUBLIC KEY-----
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDGjENfZzAK3yVGrPrQ8Qq3ZEbT
s6sOb2qOZHukWMj8EnHifveO7UAF+I133bcHrpscbCuO9vXngUtvVoIv1uZCWZLS
60/nOpxZjTxKUgJdWHFGLKdI5MKqNjgYl6fyIkOHjeqOXfjrFIOeZIlC7HIoCObu
U3GGJd4OzJjBCIw9BwIDAQAB
-----END PUBLIC KEY-----
```

第8章 常见问题

1. **Open CGI 请求参数的编码格式是怎样的？**
   每个请求参数 key 和 value 不做 url encode 编码，拼成 http post 请求串后整体做 url encode 编码；**注意：空格编码为%20，+编码为%2B。**
2. **请求参数中中文字符要怎么处理？**
   请求参数中如果有中文要做 utf-8 编码，如准实时扣款请求接口中，请求参数 in_station_name 和 out_station_name、order_desc 中如果有中文，一定要是 utf-8 编码，否则推单失败。
3. **公钥、MAC 下载下载频率是多少？**
   公钥、MAC 列表腾讯后台不会频繁变动，但为了安全和数据同步的要求，原则上要求通卡每天请求下载一次，腾讯后台修改公钥一般会新增一个索引，因此拉取到数据后直接覆盖即可。
4. **黑名单下载频率和请求时间段怎么选择？**
   黑名单为了保证数据及时同步，也要求每天下载一次，begin_time、end_time 可以选择一周 7 天的时间段的黑名单，每个 openid 也有一个有效期，实际判断按每个 openid 的有效期判断。请求 page_index 从 0 开始。
5. **何时可以请求下载对账单？**
   T+1 日 00:10 分可以下载 T 日至 T-30 日范围内的对账单，需按天下载。

第9章 附件

9.1 附录 A

9.1.1 订单类型场景

[申请扣款]的 exp_type 应用场景

当 charge_type=0 时(即一次性扫码计费)：

订单类型	名称	场景描述
0	正常	完整行程，有进、出站的扫码记录的订单 charge_data 只有 1 条进站扫码记录的计费数据

当 charge_type=1 时(即多次扫码计费)：

订单类型	名称	场景描述
0	正常	完整行程，有进、出站的扫码记录的订单 charge_data 传有进站扫码与出站扫码的计费数据
1	单边进站	只有进站扫码记录，但是超过 N 天，还是没有匹配到出站记录、BOM 或补登处理，认为为真实单边交易的订单 charge_data 只有进站扫码的计费数据
2	单边出站	只有出站扫码记录，但是超过 N 天，还是没有匹配到出站记录、BOM 或补登处理，认定为真实单边交易的订单 charge_data 只有出站扫码的计费数据
3	单边进站，客服补出站	有进站扫码记录，且在 N 天内，通过 BOM 补齐了出站记录的订单 charge_data 传有进站扫码与 BOM 出站扫码的计费数据
4	单边出站，客服补进站	有出站扫码记录，且在 N 天内，通过 BOM 补齐了进站记录的订单 charge_data 传有出站扫码与 BOM 进站扫码的计费数据
5	行程超时订单	有正常的扫码记录，但付费区停留超时的订单，订单价格是正常订单+超时罚款 charge_data 传有进站扫码与出站扫码的计费数据
6	超时罚款订单	用户在付费区停留超时，经 BOM 处理的罚款订单。用户的进出站扫码记录将形成另一笔正常订单。 charge_data 传有 BOM 扫码与进站或出站扫码的计费数据
7	进站清本站订单	用户扫码未进站，在非付费区通过 BOM 清除进站记录时，产生的订单 charge_data 只有进站扫码与 BOM 扫码的计费数据
8	补登入站(用户输入进站信息)	有出站扫码记录，且在 N 天内，通过补登功能补齐了进站记录的订单 charge_data 传有 BOM 扫码与出站扫码的计费数据
9	补登出站(用户输入出站信	有进站扫码记录，且在 N 天内，通过补登功能补齐了出站记录

	息)	的订单 charge_data 传有进站扫码与 BOM 扫码的计费数据
10	客服更新进出站状态	用户在无法进出站时，通过 BOM 更新状态后产生的订单，至少包含一条 BOM 处理记录。 charge_data 传有 1 条或 2 条 BOM 扫码的计费数据

9.1.2 后台 SDK 错误码

错误码(retcode)	错误描描述
-1	网络错误
-2	网络未连接
-3	返回数据为空
-4	验证签名失败
-5	url 错误 包含 HttpResponseCode.CODE_URL_EMPTY、 CODE_URL_ERROR
-6	socket 连接超时,请检查网络连接或 host 的配置
-7	请求签名失败
-100	请求参数错误
-101	数据解析失败

9.1.3 订单异常拦截

拦截规则

拦截类型	场景描述
3-单边账未达推送时间	单边账在约定的单边匹配时间之内推送
4-交易金额超限	扫码数据验证，交易金额超限
5-扫码记录格式错误	扫码数据验证，数据解析或验证签名失败
10-交易存在风险	推送时间距离当前时间太久，扫码数据超过推送时间范围
11-重复订单	单次刷码计费场景，两笔不同订单，扫码时间或扫码设备 id 相同 两次刷码计费场景，两笔不同订单，使用同一个二维码
12-用户欠费超限，禁止付款	用户存在未支付订单且超过腾讯规定的期限
99-系统错误，请重新申请扣款	内部网络抖动等问题，请通过查询扣款接口，确定订单是否推送成功，若未推送成功，请重新申请扣款

扫码数据校验：

核心扫码数据，判断二维码是否有效，是否过期，是否有腾讯标记，数据是否被篡改，保证资金的安全。

站点检测：

扫码数据校验只能检测二维码的合法性，无法验证请款请求本身的合法性，例如：用户进站，闸机异常，数据丢失，到 客服处 BOM 处理后出站，闸机恢复后，商户后台未处理好，又推送一笔请款过来。如果没有异常类型以及对应的站点信 息，当用户投诉时，无法反推使用场景，造成资金及产品口碑的双重损失。

推送时间校验：

防止闸机故障后，没有请款过来，过几个月后重新上线，对用户扣款，引发用户投诉。

重复订单检测：

减少因为闸机异常，或用户闸机误用、商户后台异常推单时对用户进行重复扣款，引起投诉。

9.2 机具硬件配置表

机具硬件配置表										
商户名称	主控芯片型号	内核	主频(MHz)	总内存(KB)	剩余栈内存(KB)	剩余堆内存(KB)	剩余存储容量(KB)	系统	编译环境	备注
样例商户	STM32F407ZE	CM4	168	--	5+	10+	5152+	linux	--	--

9.3 机具扫码语音及文案建议

序号	刷码结果	提示音	屏幕文案
1	成功	刷码成功/请上车	显示金额，如“1.00 元”；或者“刷码成功”
2	同一码重刷	不能重刷	不能重刷
3	二维码过期	无效码	二维码过期
4	超过最大金额		超过最大金额
5	二维码格式错误		请出示正确二维码
6	卡证书公钥错误		证书公钥错误
7	卡证书验签失败		证书验签失败
8	卡证书用户公钥错误		用户公钥错误
9	二维码验签错误		二维码验签错误
10	卡证书过期		卡证书过期
11	二维码过期		二维码过期
13	openid 错误		openid 错误
14	参数错误		参数错误
15	内存申请不足		内存申请不足
16	卡证书签名算法不支持		卡证书签名算法错误
17	加密的 mac 根密钥解密失败		mac 根密钥解密错误
18	mac 校验失败		mac 校验失败
19	二维码签名算法不支持		二维码签名算法错误
20	扫码记录加密失败		扫码记录加密失败

21	扫码记录编码失败		扫码记录编码失败
22	其它错误		系统错误

9.4 腾讯乘车码机具功能检查表

9.4.1 机具功能验收表

机具功能验收表			
类型	验证项	状态	备注
公钥	公钥下载		
	公钥存储		
黑名单	黑名单下载		
	黑名单存储(10W 条)		
MAC 密钥	MAC 密钥下载		
	MAC 密钥存储		
时间同步	与互联网时间误差小于 10s		
正常扫码	验码通过		
	验码耗时小于 300ms		
非法用户拦截	黑名单用户		
非法行为拦截	同一码重复扫码		
	同一用户 5s 内换码刷，应拦截		
交易记录	交易记录生成		
	交易记录存储(1W 条)		
推单代扣	成功推单		
	在规定时限内推单		

9.4.2 机具体验验收表

类型	验证项	验证用例	期待结果
基础功能	扫码支付	打开乘车码正常刷码	刷码成功，有支付成功弹框、聊天列表有扣款通知、乘车记录页有对应的乘车记录
稳定性	时间稳定性	相隔 1 天，再次刷码	刷码成功，同上
	机具重启	机具重启后，再次刷码	刷码成功，同上
	不同机型测试	苹果和安卓系统	刷码成功，同上

	多人连续扫码支付	5 人以上，连续刷码成功	刷码成功，同上
	不同距离识别速度	尝试不同距离（完全贴近-10 里面）是否都能识别	刷码成功，同上
	不同角度识别速度	与扫码模块成左倾、右倾、前倾、后倾不同角度	刷码成功，同上
防重刷	同一个码防重刷	同一个码，在 1 分钟内不能重刷	不能重刷
		两个用户交替刷码（一码不变，截图保存；另一个码每次更新）	码没有变的一方不能“刷码成功”
	同一个用户防重刷	刷码，同一用户 5 秒内刷新码再刷码。	不能重刷
		刷码，同一用户 5s 后刷新码再刷码。	期待结果应与业主方确认，但应支持该能力
	两个用户交替刷码防重刷	两个用户交替刷自己的同一码	不能重刷
		两个用户 1 分钟内交替刷码，刷码后均刷新二维码	期待结果应与业主方确认，但应支持该能力
扣款通知	扣款通知延迟时长	打开乘车码正常刷码	10 秒内可以看到支付成功页
机具弱网络测试	无网络刷码	机具断网刷码，网络恢复后，是否推送订单代扣	刷码成功，网络恢复后代扣成功
断电测试	机具网络异常+断电	1、机具断网 2、刷码 3、机具断电重启 4、机具恢复网络	2、刷码成功 4、网络恢复后代扣成功
压力测试	机具批量刷码，网络等原因导致订单不能及时推送	1、机具断网 2、长时间不停刷码 3、机具网络恢复	2、刷码成功 3、网络恢复后所有订单推送成功

9.5 腾讯乘车码后台检查表

腾讯乘车码后台检查表			
类型	验证项	状态	备注
公钥	公钥下载		
	公钥存储		
MAC 密钥	MAC 密钥下载		

	MAC 密钥存储		
黑名单	黑名单下载		
	黑名单存储		
交易记录	推单		
对账	定期下载对账单		
	完成对账		
	查单接口		

9.6 银行列表

字符型银行编码	银行名称
ICBC_DEBIT	工商银行(借记卡)
ICBC_CREDIT	工商银行(信用卡)
ABC_DEBIT	农业银行(借记卡)
ABC_CREDIT	农业银行(信用卡)
PSBC_DEBIT	邮政储蓄银行(借记卡)
PSBC_CREDIT	邮政储蓄银行(信用卡)
CCB_DEBIT	建设银行(借记卡)
CCB_CREDIT	建设银行(信用卡)
CMB_DEBIT	招商银行(借记卡)
CMB_CREDIT	招商银行(信用卡)
BOC_DEBIT	中国银行(借记卡)
BOC_CREDIT	中国银行(信用卡)
COMM_DEBIT	交通银行(借记卡)
COMM_CREDIT	交通银行(信用卡)
SPDB_DEBIT	浦发银行(借记卡)
SPDB_CREDIT	浦发银行(信用卡)
GDB_DEBIT	广发银行(借记卡)
GDB_CREDIT	广发银行(信用卡)
CMBC_DEBIT	民生银行(借记卡)
CMBC_CREDIT	民生银行(信用卡)
PAB_DEBIT	平安银行(借记卡)
PAB_CREDIT	平安银行(信用卡)
CEB_DEBIT	光大银行(借记卡)
CEB_CREDIT	光大银行(信用卡)
CIB_DEBIT	兴业银行(借记卡)
CIB_CREDIT	兴业银行(信用卡)
CITIC_DEBIT	中信银行(借记卡)
CITIC_CREDIT	中信银行(信用卡)
BOSH_DEBIT	上海银行(借记卡)
BOSH_CREDIT	上海银行(信用卡)
CRB_DEBIT	华润银行(借记卡)
HZB_DEBIT	杭州银行(借记卡)

HZB_CREDIT	杭州银行(信用卡)
BSB_DEBIT	包商银行(借记卡)
BSB_CREDIT	包商银行(信用卡)
CQB_DEBIT	重庆银行(借记卡)
SDEB_DEBIT	顺德农商行(借记卡)
SZRCB_DEBIT	深圳农商银行(借记卡)
SZRCB_CREDIT	深圳农商银行(信用卡)
HRBB_DEBIT	哈尔滨银行(借记卡)
BOCD_DEBIT	成都银行(借记卡)
GDNYB_DEBIT	南粤银行(借记卡)
GDNYB_CREDIT	南粤银行(信用卡)
GZCB_DEBIT	广州银行(借记卡)
GZCB_CREDIT	广州银行(信用卡)
JSB_DEBIT	江苏银行(借记卡)
JSB_CREDIT	江苏银行(信用卡)
NBCB_DEBIT	宁波银行(借记卡)
NBCB_CREDIT	宁波银行(信用卡)
NJCB_DEBIT	南京银行(借记卡)
QHNX_DEBIT	青海农信(借记卡)
ORDOSB_CREDIT	鄂尔多斯银行(信用卡)
ORDOSB_DEBIT	鄂尔多斯银行(借记卡)
BJRCB_CREDIT	北京农商(信用卡)
BHB_DEBIT	河北银行(借记卡)
BGZB_DEBIT	贵州银行(借记卡)
BEEB_DEBIT	鄞州银行(借记卡)
PZHCCB_DEBIT	攀枝花银行(借记卡)
QDCCB_CREDIT	青岛银行(信用卡)
QDCCB_DEBIT	青岛银行(借记卡)
SHINHAN_DEBIT	新韩银行(借记卡)
QLB_DEBIT	齐鲁银行(借记卡)
QSB_DEBIT	齐商银行(借记卡)
ZZB_DEBIT	郑州银行(借记卡)
CCAB_DEBIT	长安银行(借记卡)
RZB_DEBIT	日照银行(借记卡)
SCNX_DEBIT	四川农信(借记卡)
BEEB_CREDIT	鄞州银行(信用卡)
SDRCU_DEBIT	山东农信(借记卡)
BCZ_DEBIT	沧州银行(借记卡)
SJB_DEBIT	盛京银行(借记卡)
LNNX_DEBIT	辽宁农信(借记卡)
JUFENGB_DEBIT	临朐聚丰村镇银行(借记卡)
ZZB_CREDIT	郑州银行(信用卡)
JXNXB_DEBIT	江西农信(借记卡)

JZB_DEBIT	晋中银行(借记卡)
JZCB_CREDIT	锦州银行(信用卡)
JZCB_DEBIT	锦州银行(借记卡)
KLB_DEBIT	昆仑银行(借记卡)
KRCB_DEBIT	昆山农商(借记卡)
KUERLECB_DEBIT	库尔勒市商业银行(借记卡)
LJB_DEBIT	龙江银行(借记卡)
NYCCB_DEBIT	南阳村镇银行(借记卡)
LSCCB_DEBIT	乐山市商业银行(借记卡)
LUZB_DEBIT	柳州银行(借记卡)
LWB_DEBIT	莱商银行(借记卡)
LYYHB_DEBIT	辽阳银行(借记卡)
LZB_DEBIT	兰州银行(借记卡)
MINTAIB_CREDIT	民泰银行(信用卡)
MINTAIB_DEBIT	民泰银行(借记卡)
NCB_DEBIT	宁波通商银行(借记卡)
NMGNX_DEBIT	内蒙古农信(借记卡)
XAB_DEBIT	西安银行(借记卡)
WFB_CREDIT	潍坊银行(信用卡)
WFB_DEBIT	潍坊银行(借记卡)
WHB_CREDIT	威海商业银行(信用卡)
WHB_DEBIT	威海市商业银行(借记卡)
WHRC_CREDIT	武汉农商(信用卡)
WHRC_DEBIT	武汉农商行(借记卡)
WJRCB_DEBIT	吴江农商行(借记卡)
WLMQB_DEBIT	乌鲁木齐银行(借记卡)
WRCB_DEBIT	无锡农商(借记卡)
WZB_DEBIT	温州银行(借记卡)
XAB_CREDIT	西安银行(信用卡)
WEB_DEBIT	微众银行(借记卡)
XIB_DEBIT	厦门国际银行(借记卡)
XJRCCB_DEBIT	新疆农信银行(借记卡)
XMCCB_DEBIT	厦门银行(借记卡)
YNRCCB_DEBIT	云南农信(借记卡)
YRRCB_CREDIT	黄河农商银行(信用卡)
YRRCB_DEBIT	黄河农商银行(借记卡)
YTB_DEBIT	烟台银行(借记卡)
ZJB_DEBIT	紫金农商银行(借记卡)
ZJLXRB_DEBIT	兰溪越商银行(借记卡)
ZJRCUB_CREDIT	浙江农信(信用卡)
AHRCUB_DEBIT	安徽省农村信用社联合社(借记卡)
BCZ_CREDIT	沧州银行(信用卡)
SRB_DEBIT	上饶银行(借记卡)

ZYB_DEBIT	中原银行(借记卡)
ZRCB_DEBIT	张家港农商行(借记卡)
SRCB_CREDIT	上海农商银行(信用卡)
SRCB_DEBIT	上海农商银行(借记卡)
ZJTLCB_DEBIT	浙江泰隆银行(借记卡)
SUZB_DEBIT	苏州银行(借记卡)
SXNX_DEBIT	山西农信(借记卡)
SXXH_DEBIT	陕西信合(借记卡)
ZJRCUB_DEBIT	浙江农信(借记卡)
AE_CREDIT	AE(信用卡)
TACCB_CREDIT	泰安银行(信用卡)
TACCB_DEBIT	泰安银行(借记卡)
TCRCB_DEBIT	太仓农商行(借记卡)
TJBHB_CREDIT	天津滨海农商行(信用卡)
TJBHB_DEBIT	天津滨海农商行(借记卡)
TJB_DEBIT	天津银行(借记卡)
TRCB_DEBIT	天津农商(借记卡)
TZB_DEBIT	台州银行(借记卡)
URB_DEBIT	联合村镇银行(借记卡)
DYB_CREDIT	东营银行(信用卡)
CSRCB_DEBIT	常熟农商银行(借记卡)
CZB_CREDIT	浙商银行(信用卡)
CZB_DEBIT	浙商银行(借记卡)
CZCB_CREDIT	稠州银行(信用卡)
CZCB_DEBIT	稠州银行(借记卡)
DANDONGB_CREDIT	丹东银行(信用卡)
DANDONGB_DEBIT	丹东银行(借记卡)
DLB_CREDIT	大连银行(信用卡)
DLB_DEBIT	大连银行(借记卡)
DRCB_CREDIT	东莞农商银行(信用卡)
DRCB_DEBIT	东莞农商银行(借记卡)
CSRCB_CREDIT	常熟农商银行(信用卡)
DYB_DEBIT	东营银行(借记卡)
DYCCB_DEBIT	德阳银行(借记卡)
FBB_DEBIT	富邦华一银行(借记卡)
FDB_DEBIT	富滇银行(借记卡)
FJHXB_CREDIT	福建海峡银行(信用卡)
FJHXB_DEBIT	福建海峡银行(借记卡)
FJNX_DEBIT	福建农信银行(借记卡)
FUXINB_DEBIT	阜新银行(借记卡)
BOCDB_DEBIT	承德银行(借记卡)
JSNX_DEBIT	江苏农商行(借记卡)
BOLFB_DEBIT	廊坊银行(借记卡)

CCAB_CREDIT	长安银行(信用卡)
CBHB_DEBIT	渤海银行(借记卡)
CDRCB_DEBIT	成都农商银行(借记卡)
BYK_DEBIT	营口银行(借记卡)
BOZ_DEBIT	张家口市商业银行(借记卡)
CFT	零钱
BOTSB_DEBIT	唐山银行(借记卡)
BOSZS_DEBIT	石嘴山银行(借记卡)
BOSXB_DEBIT	绍兴银行(借记卡)
BONX_DEBIT	宁夏银行(借记卡)
BONX_CREDIT	宁夏银行(信用卡)
GDHX_DEBIT	广东华兴银行(借记卡)
BOLB_DEBIT	洛阳银行(借记卡)
BOJX_DEBIT	嘉兴银行(借记卡)
BOIMCB_DEBIT	内蒙古银行(借记卡)
BOHN_DEBIT	海南银行(借记卡)
BOD_DEBIT	东莞银行(借记卡)
CQRCB_CREDIT	重庆农商银行(信用卡)
CQRCB_DEBIT	重庆农商银行(借记卡)
CQTGB_DEBIT	重庆三峡银行(借记卡)
BOD_CREDIT	东莞银行(信用卡)
CSCB_DEBIT	长沙银行(借记卡)
BOB_CREDIT	北京银行(信用卡)
GDRCU_DEBIT	广东农信银行(借记卡)
BOB_DEBIT	北京银行(借记卡)
HRXJB_DEBIT	华融湘江银行(借记卡)
HSBC_DEBIT	恒生银行(借记卡)
HSB_CREDIT	徽商银行(信用卡)
HSB_DEBIT	徽商银行(借记卡)
HUNNX_DEBIT	湖南农信(借记卡)
HUSRB_DEBIT	湖商村镇银行(借记卡)
HXB_CREDIT	华夏银行(信用卡)
HXB_DEBIT	华夏银行(借记卡)
HNNX_DEBIT	河南农信(借记卡)
BNC_DEBIT	江西银行(借记卡)
BNC_CREDIT	江西银行(信用卡)
BJRCB_DEBIT	北京农商行(借记卡)
JCB_DEBIT	晋城银行(借记卡)
JJCCB_DEBIT	九江银行(借记卡)
JLB_DEBIT	吉林银行(借记卡)
JLNX_DEBIT	吉林农信(借记卡)
JNRCB_DEBIT	江南农商(借记卡)
JRCB_DEBIT	江阴农商行(借记卡)

JSHB_DEBIT	晋商银行(借记卡)
HAINNX_DEBIT	海南农信(借记卡)
GLB_DEBIT	桂林银行(借记卡)
GRCB_CREDIT	广州农商银行(信用卡)
GRCB_DEBIT	广州农商银行(借记卡)
GSB_DEBIT	甘肃银行(借记卡)
GSNX_DEBIT	甘肃农信(借记卡)
GXNX_DEBIT	广西农信(借记卡)
GYCB_CREDIT	贵阳银行(信用卡)
GYCB_DEBIT	贵阳银行(借记卡)
GZNX_DEBIT	贵州农信(借记卡)
HAINNX_CREDIT	海南农信(信用卡)
HKB_DEBIT	汉口银行(借记卡)
HANAB_DEBIT	韩亚银行(借记卡)
HBCB_CREDIT	湖北银行(信用卡)
HBCB_DEBIT	湖北银行(借记卡)
HBNX_CREDIT	湖北农信(信用卡)
HBNX_DEBIT	湖北农信(借记卡)
HDCB_DEBIT	邯郸银行(借记卡)
HEBNX_DEBIT	河北农信(借记卡)
HFB_DEBIT	恒丰银行(借记卡)
HKBEA_DEBIT	东亚银行(借记卡)
JCB_CREDIT	JCB(信用卡)
MASTERCARD_CREDIT	MASTERCARD(信用卡)
VISA_CREDIT	VISA(信用卡)
LQT	零钱通