APP 支付接入指引

1.收单链路选择指引

1.1. 普通收单链路

功能类别	具体功能	功能场景概述	交互案例示例
使用微信支付商户号完成QQ 钱包支付-C2B 收款	QQ-JSAPI 支付	QQ-JSAPI 支付是指商户通过调用微信支付提供的 JSAPI 接口，在支付场景中调起 QQ 钱包支付模块完成收款，商户的支付场景是在 **QQ 内置浏览器**，扣款方为 QQ 用户，收款方为微信支付商户号	
	QQ-APP 支付	**QQ-APP 支付是指商户通过在移动端应用 APP 中集成开放 SDK 调起 QQ 钱包支付模块来完成收款。商户的支付场景是在 APP，扣款方为 QQ 用户，收款方为微信支付商户号**	

	QQ-NATIVE 支付	QQ-Native 支付是指商户系统按微信支付协议生成支付二维码，用户再用 QQ“扫一扫”完成支付的模式。商户的支付场景是在 **QQ 客户端**，扣款方为 QQ 用户，收款方为微信支付商户号	
	QQ-H5 支付	QQ-H5 支付是指商户在 QQ 客户端外的移动端网页展示商品或服务，用户确认购买后，商户拉起 QQ 客户端进行支付的模式。商户的支付场景是在 **QQ 端外浏览器**，扣款方为 QQ 用户，收款方为微信支付商户号	
	QQ-自动续费	商户通过签约接口与用户签订委托代扣协议，签约成功后商户可以根据签约协议号发起扣款，典型场景为 QQ 会员的包月/包季/包年续费服务	-
使用微信支付商户号完成 QQ 钱包支付 -B2C 付款	商家转账	商户可以使用商家转账用于现金营销、分销返佣、行政补贴、佣金报酬、企业报销、企业补贴、服务款项、采购货款等向用户转账的场景。可将营销资金**直接转账**进 QQ 用户零钱账户	-

1.2. 电商收单链路

功能类别	具体功能	功能场景概述	交互案例示例
使用微信支付商户号完成QQ钱包支付-C2B收款	QQ-JSAPI支付	QQ-JSAPI支付是指商户通过调用微信支付提供的JSAPI接口，在支付场景中调起QQ钱包支付模块完成收款，商户的支付场景是在**QQ内置浏览器**，扣款方为QQ用户，收款方为微信支付商户号	
	QQ-APP支付	**QQ-APP支付是指商户通过在移动端应用APP中集成开放SDK调起QQ钱包支付模块来完成收款。商户的支付场景是在APP，扣款方为QQ用户，收款方为微信支付商户号**	
	QQ-NATIVE支付	QQ-NATIVE支付是指商户系统按微信支付协议生成支付二维码，用户再用QQ“扫一扫”完成支付的模式。商户的支付场景是在**QQ客户端**，扣款方为QQ用户，收款方为微信支付商户号	

	QQ-H5 支付	QQ-H5 支付是指商户在 QQ 客户端外的移动端网页展示商品或服务，用户确认购买后，商户拉起 QQ 客户端进行支付的模式。商户的支付场景是在 **QQ 端外浏览器**，扣款方为 QQ 用户，收款方为微信支付商户号	
使用微信支付商户号完成 QQ 钱包支付-分账	分账微信商户	订单支付成功后，由电商平台发起分账请求，将结算后的资金分给分账接收方，目前已支持分账电商平台子商户	-

1.3. 普通收单链路接入流程

1、 微信支付商户号申请&使用：请按照商户入驻指引完成微信支付商户号开通，或者使用已有的微信支付商户号；

2、 QQ 互联账号申请（新号需要接入）：创建 QQ 互联开发者账号与网站应用 appid，并完成 QQ 账户登陆体系接入，详见本文目录 2.1QQ 互联接入或 QQ 互联 WIKI，如有疑问请咨询 qqconnecthelper；

3、 支付权限开通：目前仅支持人工配置，请按「QQ 钱包支付商户平台商户号升级操作流程」目录 1.2 进行开通，接收需求后将于 7 个工作日内完成权限配置，并根据所需业务场景的支付方式接入指引（详见目录 3），进行接入。

1.4. 电商收付通收单链路接入流程

1、 微信支付商户号申请&使用：请按照商户入驻指引完成微信支付商户号开通，或者使用已有的微信支付商户号；

2、 收付通权限开通：开通电商收付通相关权限详见开通平台收付通的资质要求；

3、 QQ 互联账号申请（新号需要接入）：创建 QQ 互联开发者账号与网站应用 appid，并完成 QQ 账户登陆体系接入，详见本文目录 2.1QQ 互联接入或者 QQ 互联 WIKI，如有疑问请咨询 qqconnecthelper；

4、 支付权限开通：目前仅支持人工配置，请按「QQ 钱包支付商户平台商户号升级操作流程」目录 1.2 进行开通，接收需求后将于 7 个工作日内完成权限配置，并根据所需业务场景的支付方式接入指引（详见目录 3），进行接入。

2.QQ 互联接入指引

2.1. 步骤指引

登录互联官网，使用 QQ 号完成开发者注册，需申请企业类开发者；审核通过后可在互联平台创建应用，从而获取 appid、appkey，步骤及指引详见：

- 开发者注册
- 网站接入
- Server-side 模式
- 移动端 android/iOS 接入
- 移动端 sdk 下载
- openid 获取

2.2. 答疑解惑

若研发对接过程中若有疑问，可快速通过以下渠道解决：

- 可直接在线咨询：腾讯应用开放平台

腾讯应用开放平台
专业的客服支持
您在使用开放平台过程中遇到任何问题，可点击“联系客服”
题，我们会尽快为您解决
联系客服
应用宝业务支持
网页应

腾讯社区开放平台
06-09 11:38
腾讯开放平台帮助中心
注：若无法进入人工，请关闭浏览器清理缓存重新进入
移动应用
QQ登录（QQ互联）
联运网游
OPEN功能
YSDK服务
空间页游
财务支持
创业服务
请输入，按Enter直接发送消息，Command+Enter换行
发送
腾讯社区开放平台
腾讯企点提供技术支持

3.APP 支付接口调用：使用微信支付 V3 版本接口调试 QQ 钱包支付功能

3.1. 下单

商户先调用「微信支付 APP 下单接口」生成预支付交易单。

注：下单接口文档复用微信官网接口（APP 支付），整体字段无变化，部分字段枚举新增 QQ 类型，具体如下：

参数名	变量	微信接口 forQQ 钱包支付场景适配	示例
应用 ID	appid	QQ 互联的 AppID	102011123
用户标识	openid	用户在 QQ 互联 AppID 下的 openid 获取指引	odFzSjrWJAY6wH-EpDfTdREqzAlw

3.2. 调用支付接口支付

商户通过“3.1 下单”接口获取到 prepay_id（发起支付时的必要参数）后，按「QQ-APP 支付调用方式」发起支付。

QQ-APP 支付调起支付 SDK 指引见本文 3.3。

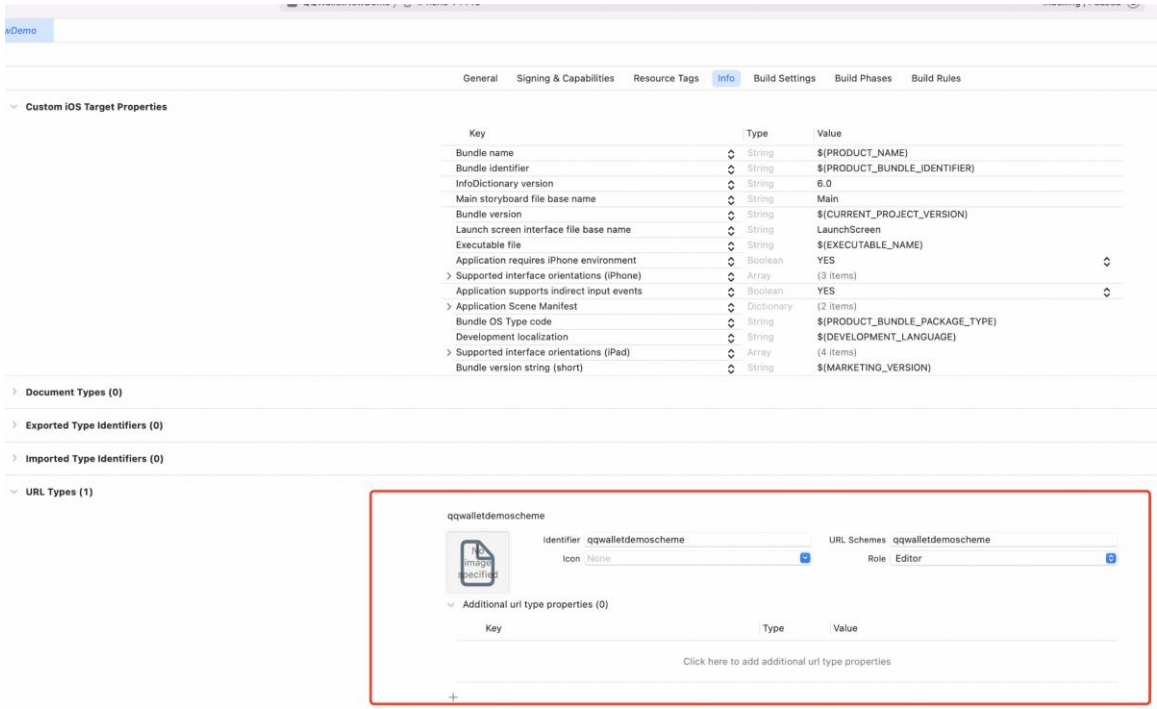
3.3. QQ-APP 支付调起支付 SDK 接入指引-IOS 端

3.3.1. 第三方 APP 内部集成 QQWalletSDK

3.3.1.1. SDK 以源码方式提供给第三方 APP，点击下载 QQWalletSDK.zip， 复制文件夹 QQWalletSDK 及内容到第三方 APP 工程中。

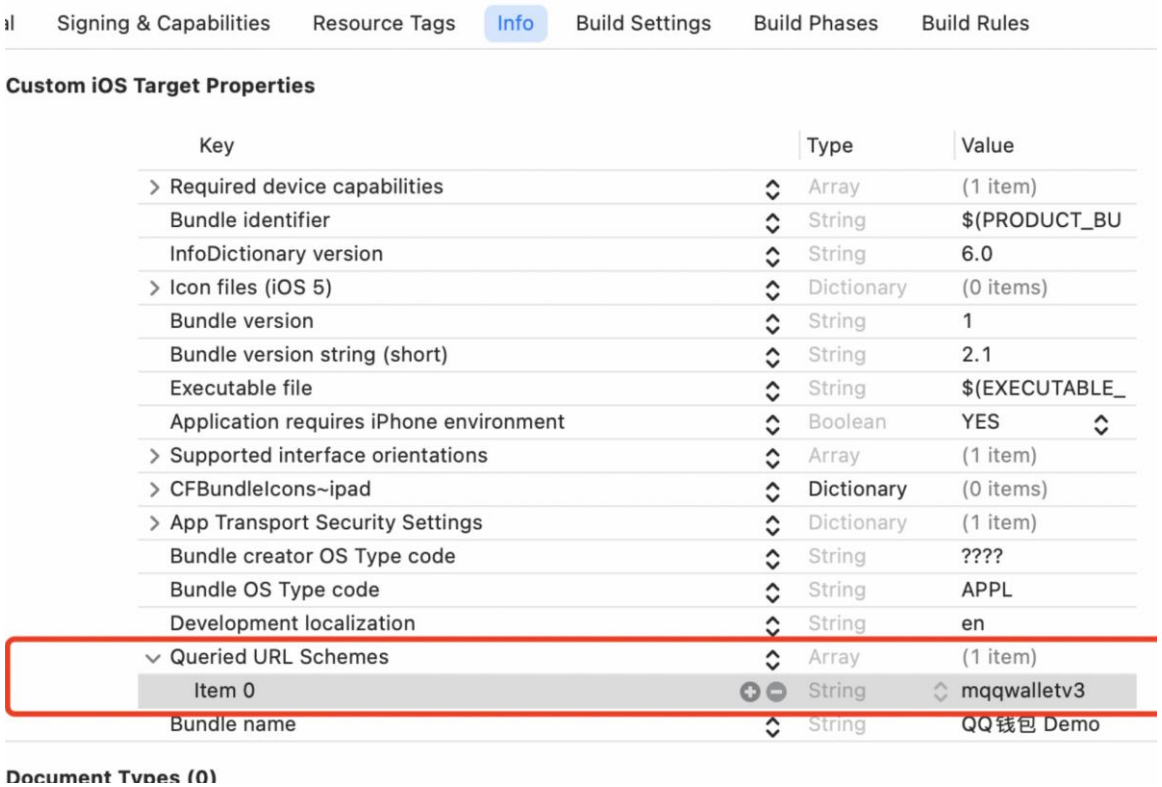
3.3.1.2. 创建用于回调的 URL Scheme

在您的工程中的 info.plist 文件中创建用于回调的 URLScheme。此 URL Scheme 用于手机 QQ 完成支付功能后，传递结果信息到第三方 APP。请尽量保证此 URL Scheme 不会与其他冲突，如下图所示



3.3.1.3. 增加 QQ 支付 Scheme 白名单

在您的工程中的 info.plist 文件中 LSApplicationQueriesSchemes 字段里增加值 mqqwalletv3，如下图所示



3.3.1.4. 监听 QQ 支付结果回调

在**appDelegate.m 中注册回调监听，如下

```
- (BOOL)application:(UIApplication *)app openURL:(NSURL *)url
options:(NSDictionary<UIApplicationOpenURLOptionsKey,id> *)options {
    // 在此函数中注册回调监听
```

```
    if ([[QQWalletSDK sharedInstance] hanldeOpenURL:url]) {
        return YES;
    }
    return YES;
}
```

3.3.2. 调用 QQ 支付接口

3.3.2.1. 支付接口

```
/// 第三方调用 QQ 钱包进行支付，新版 API
/// @param payReq 第三方向 QQ 终端发起支付的参数结构
/// @param payResult 第三方向 QQ 终端发起支付后，QQ 终端回调处理结果，包含错误码和错误信息
- (void)startPay:(QQWalletPayReq *)payReq completion:(QQWalletPayResult) payResult;
```

详细见 QQWalletSDK.h

3.3.2.2. QQWalletPayReq 定义

```
/*! @brief 第三方向 QQ 终端发起支付的消息结构体
```

```
 *
 *  第三方向 QQ 终端发起支付的消息结构体，QQ 终端处理后会向第三方返回处理结果
 *
 *  @see QQWalletPayResp
 */
@interface QQWalletPayReq : NSObject

/** 第三方 APP 在 QQ 钱包开放平台申请的 appId */
@property (nonatomic, copy) NSString *appId;

/** 商家向财付通申请的商家 id */
@property (nonatomic, copy) NSString *partnerId;

/** 预支付订单 */
@property (nonatomic, copy) NSString *prepayId;
```

```
/** 商家根据财付通文档填写的数据和签名 */
@property (nonatomic, copy) NSString *package;

/** 随机串，防重发 */
@property (nonatomic, copy) NSString *nonceStr;

/** 时间戳，防重发 */
@property (nonatomic, copy) NSString *timeStamp;

/** 商家根据开放平台文档对数据做的签名 */
@property (nonatomic, copy) NSString *sign;

/** 签名方式 */
@property (nonatomic, copy) NSString *signType;

/**透传字段，在 startPay:回调时会原样返回,最大长度 1024 字节**/
```

```
@property (nonatomic, copy) NSString *extData;

/**第三方 APP 的 bundleId*/
@property (nonatomic, copy) NSString *bundleId;

/**第三方 app 拉起 QQ 支付的鉴权签名*/
@property (nonatomic, copy) NSString *qqSign;

/**第三方 app 拉起 QQ 支付的鉴权签名算法*/
@property (nonatomic, copy) NSString *qqSignType;



@end
```

3.3.2.3. QQWalletPayResp 定义

```
/*! @brief QQ 终端返回给第三方的关于支付结果的结构体
```

```
 *
 *  QQ 终端返回给第三方的关于支付结果的结构体
 */
@interface QQWalletPayResp : NSObject
/** 错误码 */
@property (nonatomic, assign) QQWalletErrCode errCode;
/** 错误提示字符串 */
@property (nonatomic, copy) NSString *errStr;

/** 预支付订单 */
@property (nonatomic, copy) NSString *prepayId;

/**回传 QQWalletPayReq 中的 extData 字段**/
@property (nonatomic, copy) NSString *extData;
```

```
@end
```

3.3.2.4. 错误码 QQWalletErrCode 定义

```
/**
 * 错误码定义
 */
typedef enum{
    QQWalletErrCodeSuccess     = 0,          /**< 成功    */
    QQWalletErrCodeCommon      = -1,         /**< 普通错误类型    */
    QQWalletErrCodeNotSupport = -2,          /**< 当前手 Q 版本不支持 QQ 支付    */
    QQWalletErrCodeUserCancel = -11001,      /**< 用户点击取消并返回    */
} QQWalletErrCode;
```

3.4. QQ-APP 支付调起支付 SDK 接入指引-安卓端

3.4.1. APP 支付方式

调用方式

整体上跟之前保持一致，可以参考 QQ 钱包商户平台

```
IOpenApi openApi = OpenApiFactory.getInstance(context);
PayApiV2 api = new PayApiV2();
api.appId = APP_ID; // 在 QQ 开放平台注册的 AppId,参与支付签名，签名关键字 key 为 appId
api.partnerId = ...;
...
openApi.execApi(api);
```

请求对象 com.tencent.mobileqq.openpay.data.pay.PayApiV2

参数说明：

参数 1～9 跟微信保持一致，可以查阅微信支付-开发者文档，其余为 qq 特有的参数，具体看参数说明

```
/**
```

* 商户号，请填写商户号 mchid 对应的值。

* 示例值：1900000109

*/

1. **String partnerId;**

/**

* 微信返回的支付交易会话 ID，该值有效期为 2 小时。

* 示例值： WX1217752501201407033233368018

*/

2. **String prepayId;**

/**

* 随机字符串，不长于 32 位。推荐随机数生成算法。

* 示例值： 5K8264ILTKCH16CQ2502SI8ZNMTM67VS

*/

3. **String nonceStr;**

/**

* 时间戳，单位：秒，标准北京时间，时区为东八区，自 1970 年 1 月 1 日 0 点 0 分 0 秒以来的秒数。注意：部分系统取到的值为毫秒

级，需要转换成秒(10 位数字)。

* 示例值：1412000000

*/

4. String timeStamp;

/**

* 订单详情扩展字符串

* 暂填写固定值 Sign=WXPay

* 示例值：Sign=WXPay

*/

5. String packageValue;

/**

* 签名，使用字段 appId、timeStamp、nonceStr、prepayid 计算得出的签名值

* 注意：取值 RSA 格式

*/

6. String sign;

/**

```
 * 签名方式，非必填
 */
```

7. **String signType;**

```
/**
 * 透传字段，限制长度不能大于{@link #EXTDATA_MAX_LENGTH}
 */
```

8. **String extData;**

```
/**
 * 请求唯一标识
 */
```

9. **String transaction;**

```
/**
 * 腾讯 QQ 开放平台上的第三方应用 id.参与支付签名，签名关键字 key 为 appId.
 */
```

10.String appId;

/**

*APP 鉴权字段，使用 qq 互联的 appKey 对 sign、appId、packageName、callbackScheme 进行签名.QQ 钱包 APP 支付方式数字签名说明

*/

11.String qqSign;

/**

*qqSign 使用的签名方式，默认 HMAC-SHA1

*/

12.String qqSignType;

/**

*注册 qq 互联的 app 包名，也是当前 app 的 packageName

*/

13.String packageName;

```
/**
 * 手 Q 将支付结果回调给 scheme 为 callbackScheme 的 activity.
 * 回调 activity 在 AndroidManifast.xml 中申明示例如下，自定义 schema 为“mqqopenpay10000001”：
 *
 * <pre class="prettyprint">
 *     <activity
 *       android:name=".CallbackActivity"
 *       android:exported="true" \>
 *        <intent-filter\>
 *            <action android:name="android.intent.action.VIEW" />
 *            <category android:name="android.intent.category.BROWSABLE" />
 *            <category android:name="android.intent.category.DEFAULT" />
 *            <data android:scheme="mqqopenpay10000001" />
 *        </intent-filter\>
 *     </activity>
 *  </pre>
 * </p>
 */
```

14.String callbackScheme;

支付回调方式跟老接口保持一致，具体查看 QQ 钱包商户平台

支付回调结果处理

（1）在 AndroidManifest.xml 配置接支付收回的 Activity，其中 android:scheme 建议填写规则：qwallet + APP_ID。该值在”启动 QQ 钱包支付”时填写在 api.callbackScheme 中。

示例:

```
<activity
    android:name="com.pay.sample.CallbackActivity"
    android:launchMode="singleTop"
    android:exported="true" >
    <intent-filter>
      <action android:name="android.intent.action.VIEW" />
      <category android:name="android.intent.category.BROWSABLE"/>
      <category android:name="android.intent.category.DEFAULT"/>
```

```
        <data android:scheme="qwallet100619284"/>
    </intent-filter>
</activity>
```

（2）在 Activity 的 onCreate 和 onNewIntent 里调用 OpenApi 的 handleIntent 处理支付回调。

示例：

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_callback);

    openApi = OpenApiFactory.getInstance(this, appId);
    openApi.handleIntent(getIntent(), this);
}

@Override
```

```
protected void onNewIntent(Intent intent) {
    super.onNewIntent(intent);
    setIntent(intent);
    openApi.handleIntent(intent, this);
}
```

（3）在 Activity 实现支付回调响应。

示例：

```
public class CallbackActivity extends Activity implements IOpenApiListener {
@Override
public void onOpenResponse(BaseResponse response) {
    if (response == null) {
        // 不能识别的 intent
        return;
    } else {
        if (response instanceof PayResponseV2) {
```

```
// 支付回调响应
                PayResponseV2 payResponse = (PayResponseV2) response;
                if (payResponse.isSuccess()) {
// 支付成功，这个支付结果不能作为发货的依据
                }
            } else {
                // 不能识别的响应
            }
        }
}
```

回包对象 com.tencent.mobileqq.openpay.data.pay.PayResponseV2

```
/**
 * 结果代码.
 */
```

```
int retCode = RET_CODE_UNKNOW;

/**
 * 结果描述.
 */
String retMsg;

/**
 * 请求唯一标识
 */
String transaction;

/**
 * 执行的 Api 名称.
 */
String apiName;

/**
```

```
 * 执行的 Api 标识.
 */
int apiMark;

/**
 * 支付交易会话 ID
 */
String prepayId;


/**
 * 请求时的透传数据
 */
String extData;
```

sdk 地址

https://qqso.qianbao.qq.com/openpay/sdk/openpay-2.0.8.aar

3.5. QQ 钱包 APP 支付方式数字签名说明

为了保障商户利益与安全，商户 App 调用 QQ 钱包支付时启用另一套签名机制。该签名机制与“商户后台与微信商户后台的签名机制”是不同的。

1、源串构造方法

（1）将需要参与签名的所有参数按 key 进行字典升序排列。

（2）将第 1 步中排序后的参数(key=value)用&拼接起来；

（3）key 中存在大小写字母，保持大小写字母的存在。不要将 key 进行统一转换为大写或小写操作；

（4）如果 value 为空, 生成格式为“key=”，这点与后台之间签名方法是不一样的；

（5）签名原始串中，字段名和字段值都采用原始值，不进行 URL Encode。

举例：

调用某个接口，接口有如下字段：

Android：

appId、pkgName、schema、sign

实际调用接口时，各字段的值：

appId=100619284

pkgName=com.tencent.mobileqq

schema=qwallet100703379

sign=bbac33e78eb6429288969cca989bf06c

正确的签名原始串是：

appId=100619284&pkgName=com.tencent.mobileqq&schema=qwallet100703379&sign=bbac33e78eb6429288969cca989bf06c

iOS：

appId、pkgName、schema、sign

实际调用接口时，各字段的值：

appId=100619284

pkgName=com.tencent.mobileqq（值为应用的 bundle id）

schema=qwallet100703379

sign=bbac33e78eb6429288969cca989bf06c

2、 密钥构造方法

（1）在 腾讯开放平台申请 appid，并获得 appKey；

（2）构造到密钥的方式：在应用的 appkey 末尾加上一个字符的“&”，即 appkey&。

示例：

appkey 值为 d139ae6fb0175e5659dce2a7c1fe84d5

正确的密钥为：d139ae6fb0175e5659dce2a7c1fe84d5&

3、 生成签名值方法

（1）使用 HMAC-SHA1 加密算法，使用”密钥构造方法“中得到的密钥对“源串构造方法”中得到的源串进行加密（注：一般程序语言中会内置 HMAC-SHA1 加密算法的函数，例如 PHP5.1.2 之后的版本可直接调用 hash_hmac 函数）；

（2）然后将加密后的字符串进行 Base64 编码（注：一般程序语言中会内置 Base64 编码函数，例如 PHP 中可直接调用 base64_encode() 函数）；

（3）最后得到的签名值 sig 结果如下：

c6xXw0tNABhOMc869h1bfxTp9Mk=