

序言

本文档旨在向开发者描述入驻开放平台的集成工作，主要包含联合登录、支付、订单回传三个部分，涉及H5适配和服务器对接两方面，平台向开发者发布 **JS-SDK & HTTP-SDK**；JS-SDK定义了H5开发接口，提供一些native功能，帮助改善H5的用户体验；HTTP-SDK封装了服务端对接的HTTP接口，开发者只需填充业务参数，即可完成向开放平台的请求，得到业务结果；提供python/java/php/asp/c#/delph等版本。

联合登录

Step 1

在进入服务H5页面(或业务需要时)，可通过JS接口logged判断用户是否已登录，H5可将登录状态存储下来，避免后续反复判断：

```
ctk.logged({
  yes: function (res) {
    var accessToken = res.accessToken; //用户accessToken，用于获取
    userId, phone等账户信息
  },
  no: function (res) {
    console.log('用户尚未登录，请登录...');
    //可唤起登录，见Step2
  }
});
```

Step 2(未登录)

当logged接口返回未登录，在必要环节可调用JS接口login唤起登录框：

```
ctk.login({
  phone: "", //用户如果在H5页面已经填写过手机号码
  //可通过该接口传递给登录交互界面，避免重复填写
  success: function (res) {
    var accessToken = res.accessToken;
  },
  fail: function (res) {
    console.log('登录失败...');
  }
});
```

Step 3

根据用户accessToken，通过服务器接口获取用户账户信息

```
#使用HTTP-API的第一步是做全局设定(仅一次)
setDefaultAppInfo(
    appkey, #平台发放给开发者的appkey
    appsecret, #与appkey配对的appsecret
    sellerId, #用于支付的卖家id(开发者充当服务"售卖"者)
    privKey, #用于支付接口签名的RSA私钥(由平台发放给开发者，平台持有对应公钥)
    pubKey #平台公钥，用户开发者做平台服务器返回值的验签
)

#获取账户信息
user_profile = UserProfileRequest()
user_profile.accessToken = "ZjZkZmE2OWUtZTdLMi00ZG" #JS接口logged或login接口返回accessToken
try:
    res = user_profile.getResponse() #添加系统参数，签名，发送请求，解析结果，验签
    userId = res['userId'] #用户在黄页平台的稳定身份标识符
    phone = res['phone'] #用户手机号
except Exception, e:
    logging.error('啊噢，出错了...', exc_info = True)
```

支付

Step 1

当用户在H5页面点击”支付”时，调用JS接口select()，唤起支付方式选择框，平台目前支持微信和支付宝两种，均集成了native sdk，保障了支付环节的体验和转化率

```

ctk.select({
  tradeService: "com.company.product", //当前服务id, 入驻时由开发者填写
  totalFee: 98.5, //订单金额

  success: function (res) {
    var tradeStr = res.tradeStr; //js-sdk返回的用于创建支付订单的加密字符串

    //向服务端发送请求创建支付订单, 见Step2
    //调用JS接口发起支付宝或微信本地支付, 见Step3
  },
  fail: function (res) {
    console.log('用户取消...');
  }
});

```

Step 2

当用户选择支付方式后, H5页面需要通过自己的服务器向平台服务器发送”创建支付订单”的请求(限制服务端对接是为了安全), 以下是使用HTTP-SDK向平台发送请求的示例代码(python):

```

#创建订单
order_create = OrderCreateRequest()
order_create.userId = "83248134183418"
order_create.notifyUrl = "https://service.provider.com/pay/callback" #异步通知地址
order_create.tradeNo = "7a03d3201c9a4b978b7834e4e873c156" #服务商系统中的交易号
order_create.tradeService = "com.company.product" #当前服务id
order_create.tradeName = "话费充值100元" #交易主题
order_create.totalFee = 98.5 #交易金额
order_create.tradeStr = "WP58v5NfB+3Eo72KBAabGw==" #js接口select返回的tradeStr
try:
    res = order_create.getResponse() #添加系统参数, 签名, 发送请求, 解析结果, 验签
    transactionId = res['transactionId'] #平台交易号
    tradeStatus = res['tradeStatus'] #交易状态, 一般为TRADE_CREATE(待支付)
    payStr = res['payStr'] #用于本地支付的加密串

    #处理订单信息, 并将得到的payStr返回给H5页面
except Exception, e:
    logging.error('啊噢, 出错了...', exc_info = True)

```

Step 3

订单创建完毕, 回到H5页面, 调用JS接口pay(payStr):

```

ctk.pay({
    payStr: "eyJwYXltZW50VHlwZSICJqaWmFzZGwifQ=="; //Step2 服务器返回的payStr
    finish: function (resCode) {
        console.log(resCode);
    },
});

```

Step 4

当服务端收到来自平台的异步通知支付结果时, 将请求body传递给OrderResultHandler, 该接口会解析、验签并将业务结果以JSON形式返回


```

#解析支付结果
order_result_handler = OrderResultHandler(body)
try:
    res = order_result_handler.parse()
    transactionId = res['transactionId']
    tradeStatus = res['tradeStatus']
    #其他交易参数(完整信息)...

    #处理订单结果
except Exception, e:
    logging.error('啊噢, 出错了...', exc_info = True)

```

订单

当订单创建或发生关键状态变更时, 通过订单回传接口告知平台:

```

order_push = OrderPushRequest()
order_create.userId = "83278472384"
order_push.orderId = "7a03d3201c9a4b978b7834e4e873c156" #服务商系统中的交易号
order_push.orderService = "com.company.product" #当前服务id
order_push.orderTitle = "18601616860 充值100元 ¥99.50" #订单中心展示主题
order_push.orderShortInfo = "2015-02-28" #订单中心展示附属信息
order_push.orderStatus = "充值成功" #订单状态
order_push.orderCreateTime = "20150528100000" #订单创建时间
order_push.orderUrl = "http://service.provider.com/order?id= 7a03d"
#订单详情H5链接
order_push.TradeStatus = "TRADE_FINISH" #订单交易状态(如果含交易)
order_push.getResponse()
try:
    res = order_push.getResponse() #添加系统参数, 签名, 发送请求, 解析结果, 验签
    assert(res == 'SUCCESS')
except Exception, e:
    logging.error('出错了, 请稍后重试', exc_info = True)

```