

ONE-PIECE支付平台接入文档

Nov./27/2018 v.2.1.1

关于本文档

本文档为ONE-PIECE支付系统的开发接入文档。本文档将详细介绍支付系统的接入步骤，方法以及不同接入方法的具体要求。

如有疑问，请联系：

邮件：onepiece.payment@gmail.com

QQ：3273008214

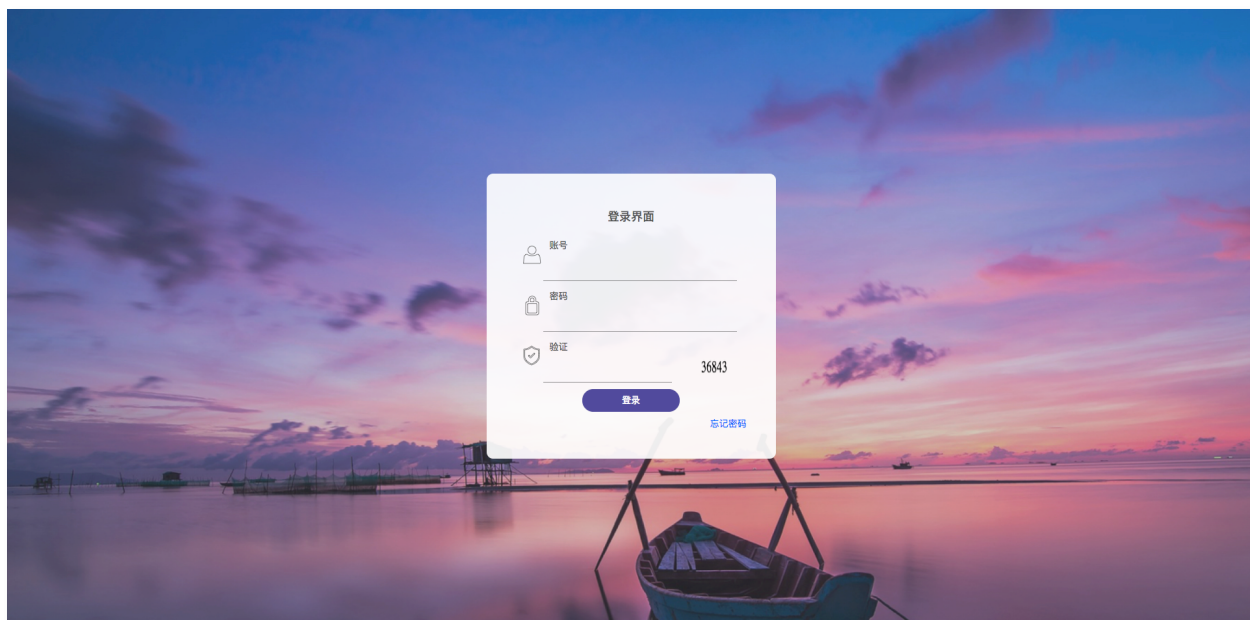
QQ：3059642261

接入步骤

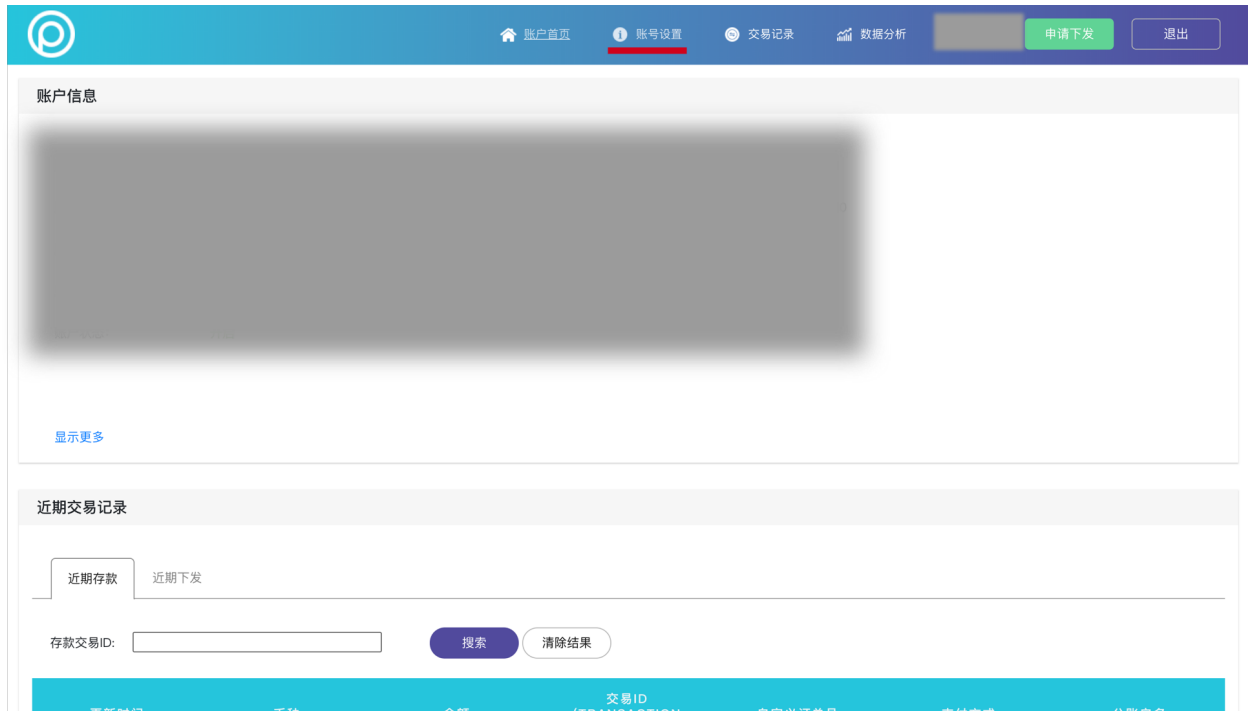
1. 创建用户账号，新用户请与我们联系：onepiece.payment@gmail.com
2. 使用步骤1. 中创建的账户，登陆控制面板：<https://admin.one-piece.us>
3. 在控制面板中创建一组新的pirate_token与pirate_key。创建pirate_token与pirate_key的详细步骤请参见本文档**创建pirate_token与pirate_key**章节。
4. 使用步骤3创建的pirate_token与pirate_key，从我们提供的两种接入方式中选择一种接入我们的服务。

创建pirate_token与pirate_key

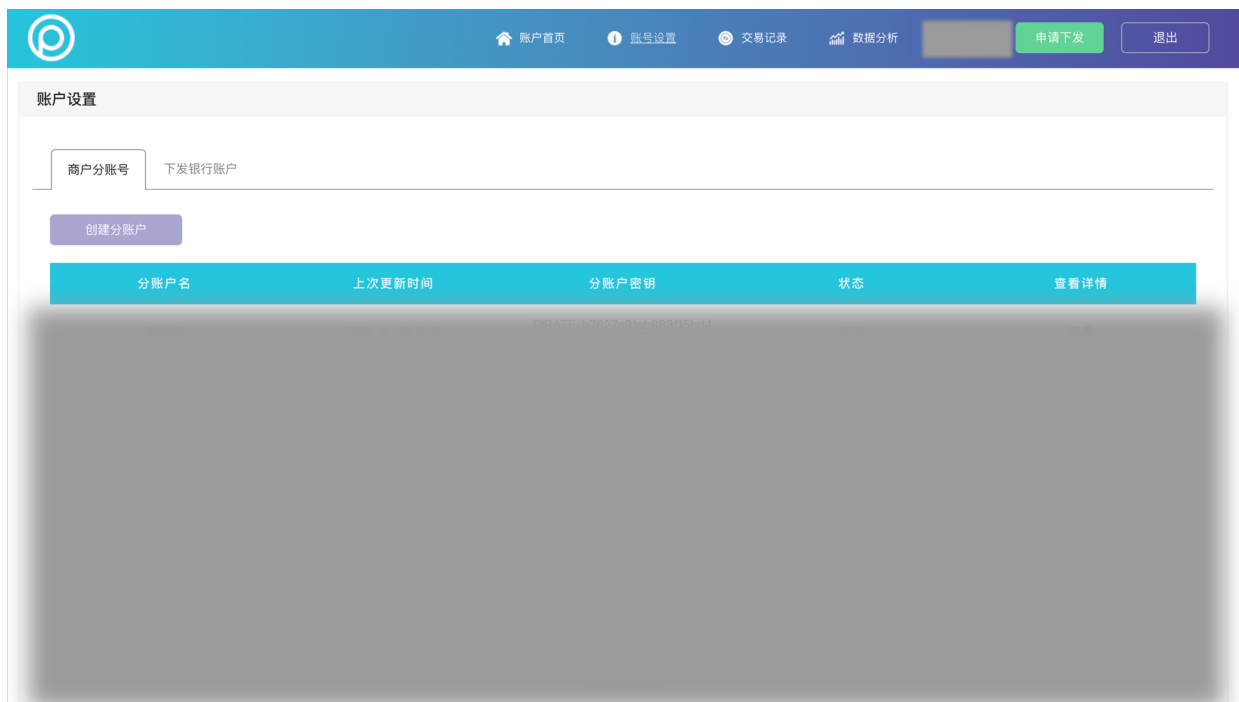
登录控制面板



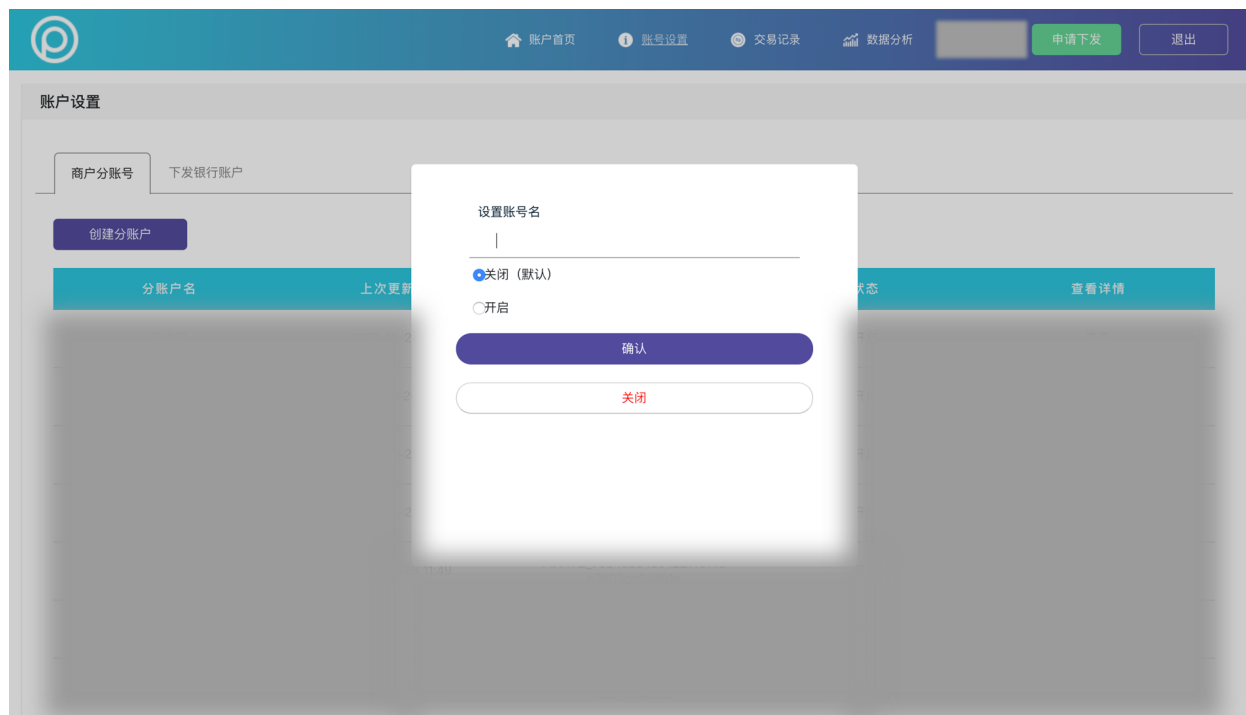
点击右上角账号设置按钮：



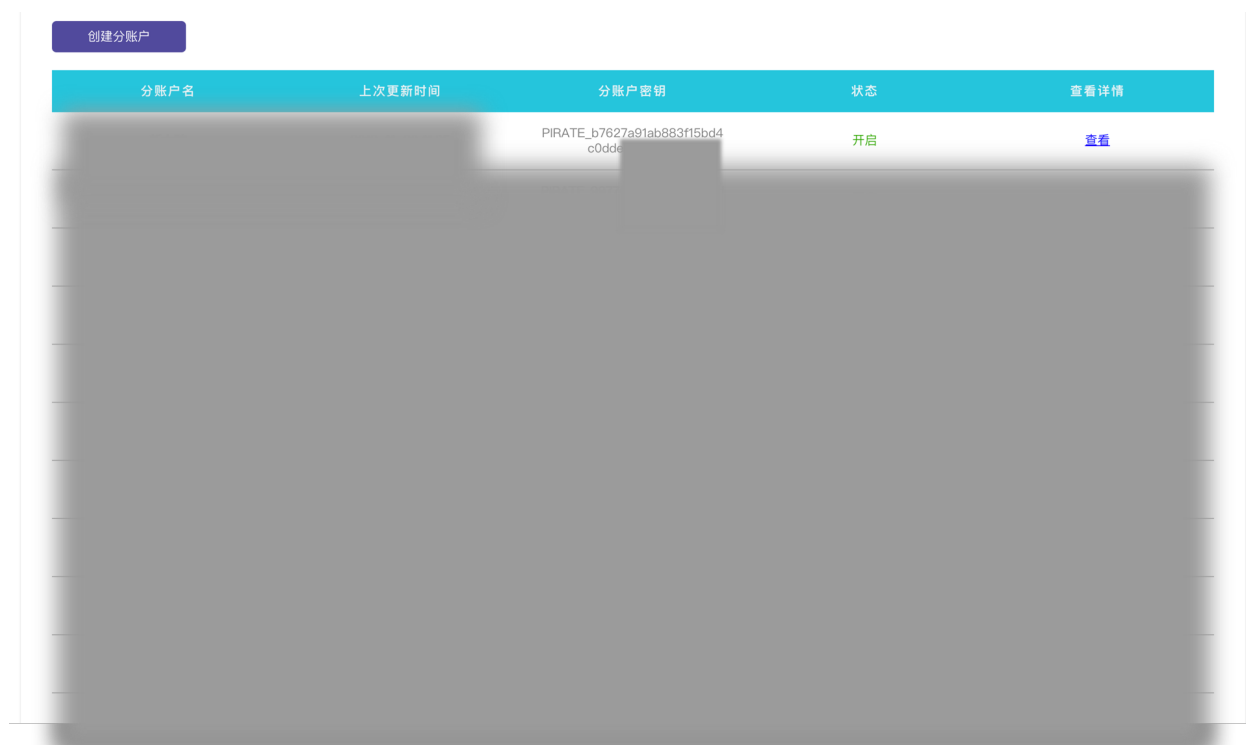
点击左上角创建分账户按钮：




按弹窗提示创建分账户：



成功创建分账户后，分账户密钥栏目下字符串即为pirate_token。要获取pirate_key，点击查看详情栏目下的查看按钮：



点击签名KEY条目下的[点击获取](#)按钮，获取pirate_key：



[账户首页](#)[账号设置](#)[交易记录](#)[数据分析](#)[申请下发](#)[退出](#)

分账户存款详细

分账户名

密钥

签名KEY[点击获取](#)

状态

更新时间

创建时间

已存款

[确认](#)[取消](#)[删除](#)

分账户成功存款记录

存款交易ID: [搜索](#)[清除结果](#)

接入方式1: 使用npm op-sdk

关于op-sdk

op-sdk npm包是one-piece支付系统的SDK包。它是一个公共npm包，所有人都可通过下方提供的链接下载并安装。本包必须配合pirate_token才能正常使用。使用op-sdk npm包您可以获取可用的付款方式列表，可用的付款金额列表，发起付款以及查询交易详情。

[op-sdk官方npm链接](#)

开始

运行下述命令安装最新版本SDK

```
//bash  
npm install op-sdk --save
```

创建OPSdk实例

```
//javascript  
import OPSdk from 'op-sdk';  
const newMerchant = new OPSdk(pirate_token) //“pirate_token”可在控制面板获取
```

SDK中包含的方法

简述	方法名	参数	描述
获取可用支付方式列表	findPaymentMethods()	-	呼叫此方法来获得当前可用的支付方式列表。
获取可用支付金额列表	findAvailablePriceList()	-	呼叫此方法来获得当前可用的支付金额列表。
发起支付	initPayment()	options	呼叫此方法来发起一笔新的付款。它有一个必填参数 options，参数类型为JavaScript Object。options参数的详细定义请参照本文档的请求参数章节。一旦一笔新的付款被成功发起，我们将在系统中创建一条新的付款记录，并返回给您 paymentToken以及付款二维码链接。用户成功付款后，系统将会自动跳转到参数中所定义的回调页面(return_url)。所有交易记录及详情可在控制面板中浏览。
查询交易状态	checkPaymentStatus()	payment_token	默认情况下，在用户成功付款后系统将发一条POST请求到参数中定义的通知地址(notify_url)，通知您付款状态。如用户希望手动查询某特定交易详情，可调用此方法查询。此方法有一个必填参数payment_token，参数类型为String。方法将返回 payment_token所对应交易状态。您还可以在控制面板中浏览全部交易状态。

React应用实例

您可参考以下应用了op-sdk的React应用实例：

<http://demo.one-piece.us>

[应用实例源代码](#)

在Node应用调用SDK

要在您的Node应用中使用op-sdk，您需要require SDK以及设置route

```
//JavaScript

//require op-sdk
const OPSdk = require('op-sdk');

//创建新的Opsdk实例
let newMerchant = new Opsdk(pirate_token) //"pirate_token"可从控制面板获取

//示例route： 获取可用支付方式
router.get('/methods', async (req, res) => {

  const getAvailableMethods = await newMerchant.findPaymentMethods();
  const availableMethods = getAvailableMethods.payload.methods;

  //Logging返回数据
  console.log(availableMethods);
  res.render('index', { title: "Available Methods", data: availableMethods })
})

//示例route： 获取可用价格列表
router.get('/price', async (req, res) => {

  const getAcceptablePriceList = await newMerchant.findAvailiblePriceList();
  const priceList = getAcceptablePriceList.payload.prices;

  //Logging返回数据
  console.log(priceList);
  res.render('index', { title: "Available Prices", data: priceList })
})

//示例route： 查询交易状态
router.get('/status/:payment_token', async (req, res) => {

  const gotStatus = await
  newMerchant.checkPaymentStatus(req.params.payment_token);
  res.render('index', { title: "Got Payment Status", data: [gotStatus.status,
  gotStatus.payload.payment_status.message ]});
});
```



```

//示例route: 发起付款
router.get('/checkout', async (req, res) => {

  const amount = (parseFloat(amount) * 100).toFixed(0); // ¥ 1需被转换为100,
  ¥ 100需被转换为10000
  const notify_url = http://yourcompany/notify/wechat //示例
  const return_url = http://yourcompany/pay/success //示例

  const options = {
    'amount': amount,
    'payment_method': 'wechatpay', //付款方式必须被拼写为: 'wechatpay',
    'alipay' or 'qqpay'
    'notify_url': notify_url,
    'return_url': return_url,
    'browser_ip_address': 'provideCustomerIP', //选填
    'browser_mac_address': 'provideCustomerMacAddress', //选填
  };

  const newTransaction = await newMerchant.initPayment(options);

  //跳转至二维码付款页面。一旦用户付款成功, 系统将会自动跳转到回调页面
  //Redirect to our QR code page. Once payment has been received, the system
  will automatically redirect to the provided "return_url"
  res.redirect(newTransaction.qrcodeURL)
});

module.exports = router;

```

请求参数

OPSdk(op-sdk构造器)

参数	必填	类型	示例数据	描述
pirate_token	是	string(145)	PIRATE_b956db50a8ffa c2d82a253a28259d07f	该数据可从控制面板获取

发起付款

参数	必填	类型	示例数据	描述
amount	是	int	100	100=1元, 10000=100元
payment_method	是	string(32)	wechatpay	付款方式必须拼写为: wechatpay/alipay/qqpay
notify_url	是	string(200)	http://yourcompany.com/notify/wechat	用户成功付款后, 系统将发送POST请求到此地址, 通知付款状态
return_url	是	string(200)	http://yourcompany.com/pay/success	付款交易完成后, 将自动跳转到此地址
browser_ip_address	否	string(65)	93.242.53.21	用户ip地址
browser_mac_address	否	string(65)	00-14-22-01-23-45	用户mac地址

查询交易状态

参数	必填	类型	示例数据	描述
payment_token	是	string(145)	TRANS_4b2107c69eb522be74c90cbbdcd1064c	交易单号

接入方式2: 呼叫我们的api

查询可用付款方式

描述: 呼叫此方法来获得当前可用的支付方式列表。

URL: https://api.one-piece.us/payment/methods/availability/{pirate_token}/{magic_num}/{signature}

Method: GET

参数:

参数	描述
pirate_token	从控制面板获取

参数	描述
magic_num	随机生成的一组数字
signature	md5(`\${magic_num}\${pirate_token}\${pirate_key}`)

查询可用支付金额

描述：呼叫此方法来获得当前可用的支付金额列表。

URL: https://api.one-piece.us/payment/prices/{pirate_token}/{magic_num}/{signature}

Method: GET

参数:

参数	描述
pirate_token	从控制面板获取
magic_num	随机生成的一组数字
signature	md5(`\${magic_num}\${pirate_token}\${pirate_key}`)

发起付款

描述：呼叫此方法来发起一笔新的付款。它有一个必填参数options，参数类型为JSON Object。options参数的详细定义请参照本文档options内容小节。一旦一笔新的付款被成功发起，我们将在系统中创建一条新的付款记录，并向参数中所定义的notify_url发起通知，通知付款交易已创建。同时返回给您paymentToken以及付款二维码链接。用户成功付款后，系统将会自动跳转到参数中所定义的回调页面(return_url)。所有交易记录及详情可在控制面板中浏览。

URL: <https://api.one-piece.us/payment/>

Method: POST

Body: {options}

options内容：

参数	必填	类型	示例数据	描述
amount	是	int	100	100=1元，10000=100元
payment_method	是	string(32)	wechatpay	付款方式必须拼写为： wechatpay/alipay/qqpay
pirate_token	是	string(145)	PIRATE_s93utkjsks56i4fu3gh6sm6sit	pirate_token可从控制面板获取
notify_url	是	string(200)	http://yourcompany.com/notify/wechat	用户成功付款后，系统将发送POST请求到此地址，通知付款状态
return_url	是	string(200)	http://yourcompany.com/pay/success	付款交易完成后，将自动跳转到此地址

参数	必填	类型	示例数据	描述
browser_ip_address	否	string(65)	93.242.53.21	用户ip地址
browser_mac_address	否	string(65)	00-14-22-01-23-45	用户mac地址
magicNum	是	int	888	一组随机数字
order_num	否	string(100)	_some_order_num_123_	自定义订单号
signature	是	string(32)	id83ud84ufje73h skd93hr5ghs83	md5(`\${magicNum}\${amount}\${payment_method}\${pirate_token}\${order_num ''}\${notify_url}\${return_url}\${pirate_key}`)

Response:

```
{
  status: true,
  payload: {
    qrcode_url: _payment_qrcode_url_,
    payment_token: _payment_token_
  }
}
```

交易状态通知

描述: 当付款交易被创建以及付款提供商返回交易结果后，系统将发起一条通知到用户提供的 notify_url，通知交易生成或交易结果。

URL: notify_url

Method: POST

Body: {
 transaction_status: false/true,
 message: TRANSACTION CREATED/TRANSACTION COMPLETED,
 payment_token: _payment_token_,
 amount: 125,
 order_num: _some_order_num_,
 signature: _signature_
}

Signature: md5(`\${transaction_status}\${message}\${payment_token}\${amount}\${pirate_token}\${order_num || ''}\${pirate_key}`)

Note: transaction_status指代付款交易是否被完成。当付款交易被首次创建，transaction_status为false。当付款交易完成，transaction_status为true。

查询交易状态

描述：默认情况下，在用户成功付款后系统将发一条POST请求到参数重定义的通知地址(notify_url)，通知用户付款成功。如用户希望手动查询某特定交易详情，可调用此方法查询。方法将返回payment_token所对应订单状态。您还可以在控制面板中浏览全部订单状态。

URL: https://api.one-piece.us/payment/status/{pirate_token}/{payment_token}/{magic_num}/{signature}

Method: GET

参数：

参数	描述
pirate_token	可从发起付款返回值中获取
magic_num	随机生成的一组数字
signature	md5(`\${magic_num}\${pirate_token}\${payment_token}\${pirate_key}`)

Response:

```
{
  status: true,
  payload: {
    payment_status: {
      complete: true,
      success: true,
      ruby: {
        transaction_id: TRANS_cab1f65ee1fde6285c8238XXXXXXXXXX,
        amount: 125,
        currency: CNY,
      }
    }
  },
  order_num: _some_order_num_123_,
  payment_token: _some_payment_token_
}
```

参考

React应用示例: <http://demo.one-piece.us/>

React应用示例源代码: <https://github.com/onepiece-payment/react-demo>

控制面板: <https://admin.one-piece.us>

如有疑问:

邮件: onepiece.payment@gmail.com

QQ: 3273008214 或 3059642261