

 免费电子书

学习

# PayPal

Free unaffiliated eBook created from  
**Stack Overflow contributors.**

#paypal

.....	1
<b>1: PayPal</b> .....	<b>2</b>
.....	2
.....	2
Examples .....	2
ID / .....	2
.....	3
<b>2: /</b> .....	<b>5</b>
.....	5
.....	5
Examples .....	5
2 .....	5
1 .....	8
<b>3: PayPal /</b> .....	<b>10</b>
.....	10
.....	10
Examples .....	10
AndroidPayPal / .....	10
<b>4:</b> .....	<b>13</b>
.....	13
Examples .....	13
Android1 .....	13
Android2 .....	14
Android3 .....	16
<b>5:</b> .....	<b>18</b>
.....	18
.....	18
Examples .....	18
ngrokExpressSandbox Webhooks .....	18
URLWebhook .....	21
<b>6: PayPal</b> .....	<b>23</b>

.....	23
.....	23
Examples.....	23
Node Express.....	23
<b>7:</b> .....	<b>26</b>
.....	26
.....	26
Examples.....	26
.....	26
.....	29
.....	<b>31</b>

---

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [paypal](#)

It is an unofficial and free PayPal ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official PayPal.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to [info@zzzprojects.com](mailto:info@zzzprojects.com)

# 1: PayPal

- PayPal API◦

1.0.0 2016411

## Examples

ID /

PayPal APIID◦

<https://developer.paypal.com/developer/applications/> “”

### REST API apps

Create an app to receive REST API credentials for testing and live transactions.

**Note** Features available for live transactions are listed in your [account eligibility](#).

Create App



#### App name

My test app

My test app 1

MyLiveApp

“”◦

## Application Details

### App Name

### Sandbox developer account

As a reminder, all apps created under your account should be related to your business and t  
By clicking the button below, you agree to [PayPal Developer Agreement](#).

Create App

ID

## SANDBOX API CREDENTIALS

Sandbox account test232213@testing.com

Client ID ATwkJTgxN3

Secret [Hide](#)

**Note:** There can only be a maximum of two client-secrets. These client-secrets can be in eit

Created	Secret
Apr 11, 2016	EJqSRO4Gj5s

Generate New Secret

PayPal API.

PayPal.

# Sandbox Test Accounts

Questions? Check out the [Testing Guide](#). Non-US developers should read our [FAQ](#).


Want to link existing Sandbox Account with your developer account? [Click Here](#) and provide

Total records: 15

<input type="checkbox"/>	Email Address	Type
<input type="checkbox"/>	▶ resttest@testing.com	PERSONAL
<input type="checkbox"/>	▶ testmctest@test.com	PERSONAL

“”。

“”。

<input type="checkbox"/>	▼ testmctest@test.com	PERSONAL
	<a href="#">Profile</a>   <a href="#">Notifications</a>	

“”。

API。

PayPal <https://riptutorial.com/zh-CN/paypal/topic/406/paypal>

## 2: /

billingAgreementAttributes	
billingPlan	ID
billingPlanAttribs	
billingPlanUpdateAttributes	
clientId	IDOAuth
HTTP	http
isoDate	ISO
	HATEOASURLPayPal
PARAMS	
	PayPal SDK
OAuth	
	PayPal

PayPal/.

1. . .
2. .
3. ID.
4. PayPal. .
5. .

## Examples

2

. ID.

3PayPal

1. ID.
2. PayPalPayPal. PayPal.
3. PayPal.



## ExpressHTTP。

- SDK body-parserJSON httpexpress ◦ IDSDKbodyParserJSON。

```
var paypal = require('paypal-rest-sdk'),
    bodyParser = require('body-parser'),
    http = require('http'),
    app = require('express')();

var clientId = 'YOUR APPLICATION CLIENT ID';
var secret = 'YOUR APPLICATION SECRET';

paypal.configure({
  'mode': 'sandbox', //sandbox or live
  'client_id': clientId,
  'client_secret': secret
});

app.use(bodyParser.json());
```

## PayPal。IDIDURL

```
http://localhost:3000/createagreement?plan=P-3N543779E9831025ECYGDNVQ
```

◦

```
app.get('/createagreement', function(req, res){
  var billingPlan = req.query.plan;

  var isoDate = new Date();
  isoDate.setSeconds(isoDate.getSeconds() + 4);
  isoDate.toISOString().slice(0, 19) + 'Z';

  var billingAgreementAttributes = {
    "name": "Standard Membership",
    "description": "Food of the World Club Standard Membership",
    "start_date": isoDate,
    "plan": {
      "id": billingPlan
    },
    "payer": {
      "payment_method": "paypal"
    },
    "shipping_address": {
      "line1": "W 34th St",
      "city": "New York",
      "state": "NY",
      "postal_code": "10001",
      "country_code": "US"
    }
  };

  // Use activated billing plan to create agreement
  paypal.billingAgreement.create(billingAgreementAttributes, function (error,
  billingAgreement){
    if (error) {
      console.error(error);
    }
  });
});
```

```

        throw error;
    } else {
        //capture HATEOAS links
        var links = {};
        billingAgreement.links.forEach(function(linkObj) {
            links[linkObj.rel] = {
                'href': linkObj.href,
                'method': linkObj.method
            };
        });

        //if redirect url present, redirect user
        if (links.hasOwnProperty('approval_url')){
            res.redirect(links['approval_url'].href);
        } else {
            console.error('no redirect URI present');
        }
    }
});
});

```

ID.

billingAgreementAttributes. ID.

billingAgreement.create(...) billingAgreementAttributes. HATEOAS. approval\_url.

. approval\_url PayPal.

PayPalURL. URLPayPal. .

.

```

app.get('/processagreement', function(req, res){
    var token = req.query.token;

    paypal.billingAgreement.execute(token, {}, function (error, billingAgreement) {
        if (error) {
            console.error(error);
            throw error;
        } else {
            console.log(JSON.stringify(billingAgreement));
            res.send('Billing Agreement Created Successfully');
        }
    });
});

```

billingAgreement.execute. . .

HTTP.

```

//create server
http.createServer(app).listen(3000, function () {
    console.log('Server started: Listening on port 3000');
});

```

# 1

◦ ◦

## PayPal Node SDK ◦ NPM

```
npm install paypal-rest-sdk
```

### .jsSDKSDKIDSDK◦

```
var paypal = require('paypal-rest-sdk');

var clientId = 'YOUR CLIENT ID';
var secret = 'YOUR SECRET';

paypal.configure({
  'mode': 'sandbox', //sandbox or live
  'client_id': clientId,
  'client_secret': secret
});
```

### JSON◦ billingPlanAttribs billingPlanUpdateAttributes◦

```
var billingPlanAttribs = {
  "name": "Food of the World Club Membership: Standard",
  "description": "Monthly plan for getting the t-shirt of the month.",
  "type": "fixed",
  "payment_definitions": [{
    "name": "Standard Plan",
    "type": "REGULAR",
    "frequency_interval": "1",
    "frequency": "MONTH",
    "cycles": "11",
    "amount": {
      "currency": "USD",
      "value": "19.99"
    }
  }
}],
  "merchant_preferences": {
    "setup_fee": {
      "currency": "USD",
      "value": "1"
    },
    "cancel_url": "http://localhost:3000/cancel",
    "return_url": "http://localhost:3000/processagreement",
    "max_fail_attempts": "0",
    "auto_bill_amount": "YES",
    "initial_fail_amount_action": "CONTINUE"
  }
};

var billingPlanUpdateAttributes = [{
  "op": "replace",
  "path": "/",
  "value": {
    "state": "ACTIVE"
  }
}];
```

```
    }  
  }];  
};
```

billingPlanAttribs

- **name / description / type** ◦
- **payment\_definitions** ◦ ◦
- **merchant\_preferences URL** ◦ ◦

◦

```
paypal.billingPlan.create(billingPlanAttribs, function (error, billingPlan){  
  if (error){  
    console.log(error);  
    throw error;  
  } else {  
    // Activate the plan by changing status to Active  
    paypal.billingPlan.update(billingPlan.id, billingPlanUpdateAttributes, function(error,  
response){  
      if (error) {  
        console.log(error);  
        throw error;  
      } else {  
        console.log(billingPlan.id);  
      }  
    }  
  });  
}  
});
```

billingPlan.create(...) billingPlanAttribs ◦ ◦ **ID** ◦

billingPlan.update(...) **ID** billingPlanUpdateAttributes ◦ ◦

**ID** billingPlan.id ◦

◦

[/ https://riptutorial.com/zh-CN/paypal/topic/467/-](https://riptutorial.com/zh-CN/paypal/topic/467/)

## 3: PayPal /

	PayPalID
	PayPal
paymentConfig	
serviceConfig	

### Examples

#### AndroidPayPal /

[PayPal Android SDK](#) [PayPal](#) [PayPal](#)

[PayPal Developer Github](#)

◦

[SDK](#) [build.gradle](#)

```
dependencies {  
    compile 'com.paypal.sdk:paypal-android-sdk:2.14.1'  
    ...  
}
```

MainActivity.java [PayPalID](#) [config](#)

```
private static PayPalConfiguration config = new PayPalConfiguration()  
    .environment(PayPalConfiguration.ENVIRONMENT_SANDBOX)  
    .clientId("YOUR CLIENT ID");
```

onCreate(...) [PayPal](#)

```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
  
    final Button button = (Button) findViewById(R.id.paypal_button);  
}
```

◦ `res> layout> main XML` `paypal_button ID` `onClick`

```
<Button android:id="@+id/paypal_button"
```

```

android:layout_height="wrap_content"
android:layout_width="wrap_content"
android:text="@string/paypal_button"
android:onClick="beginPayment" />

```

beginPayment(...) ◦ strings.xml

```
<string name="paypal_button">Pay with PayPal</string>
```

◦ onCreate(...)beginPayment(...) onCreate(...)◦

```

public void beginPayment(View view){
    Intent serviceConfig = new Intent(this, PayPalService.class);
    serviceConfig.putExtra(PayPalService.EXTRA_PAYPAL_CONFIGURATION, config);
    startService(serviceConfig);

    PayPalPayment payment = new PayPalPayment(new BigDecimal("5.65"),
        "USD", "My Awesome Item", PayPalPayment.PAYMENT_INTENT_SALE);

    Intent paymentConfig = new Intent(this, PaymentActivity.class);
    paymentConfig.putExtra(PayPalService.EXTRA_PAYPAL_CONFIGURATION, config);
    paymentConfig.putExtra(PaymentActivity.EXTRA_PAYMENT, payment);
    startActivityForResult(paymentConfig, 0);
}

```

IDconfig serviceConfig ◦ ◦ ◦ ◦ paymentConfig configpayment◦

PayPalPayPalcard.io◦

PayPal◦ onActivityResult(...)◦

```

@Override
protected void onActivityResult (int requestCode, int resultCode, Intent data){
    if (resultCode == Activity.RESULT_OK){
        PaymentConfirmation confirm = data.getParcelableExtra(
            PaymentActivity.EXTRA_RESULT_CONFIRMATION);
        if (confirm != null){
            try {
                Log.i("sampleapp", confirm.toJSONString().toString(4));

                // TODO: send 'confirm' to your server for verification

            } catch (JSONException e) {
                Log.e("sampleapp", "no confirmation data: ", e);
            }
        }
    } else if (resultCode == Activity.RESULT_CANCELED) {
        Log.i("sampleapp", "The user canceled.");
    } else if (resultCode == PaymentActivity.RESULT_EXTRAS_INVALID) {
        Log.i("sampleapp", "Invalid payment / config set");
    }
}

```

onActivityResult(...)resultCode RESULT\_OKRESULT\_CANCELEDRESULT\_EXTRAS\_INVALID

◦ ◦

```

{
  "client": {
    "environment": "sandbox",
    "paypal_sdk_version": "2.14.1",
    "platform": "Android",
    "product_name": "PayPal-Android-SDK"
  },
  "response": {
    "create_time": "2016-05-02T15:33:43Z",
    "id": "PAY-0PG63447RB821630KK1TXGTY",
    "intent": "sale",
    "state": "approved"
  },
  "response_type": "payment"
}

```

responsestateapproved state ◦ ◦ ◦

onDestroy(...)◦

```

@Override
public void onDestroy(){
    stopService(new Intent(this, PayPalService.class));
    super.onDestroy();
}

```

## ◦ PayPal◦

- beginPayment(...)beginPayment(...)◦
- ◦
- ◦

PayPal / <https://riptutorial.com/zh-CN/paypal/topic/608/paypal-->

# 4:

AndroidPayPal◦

## Examples

### Android1

PayPal Developer GithubAndroid +◦

AndroidNode◦

PayPalConfiguration◦

```
private static PayPalConfiguration config = new PayPalConfiguration()
    .environment(PayPalConfiguration.ENVIRONMENT_SANDBOX)
    .clientId("YOUR APPLICATION CLIENT ID")
    .merchantName("My Store")
    .merchantPrivacyPolicyUri(Uri.parse("https://www.example.com/privacy"))
    .merchantUserAgreementUri(Uri.parse("https://www.example.com/legal"));
```

onCreate(...).◦ ◦

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

    final Button button = (Button) findViewById(R.id.paypal_button);
}
```

res > layout > activity\_main.xmlbeginFuturePayment(...).◦

```
<Button android:id="@+id/paypal_button"
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:text="@string/paypal_button"
    android:onClick="beginFuturePayment" />
```

res > values > strings.xml◦

```
<string name="paypal_button">Process Future Payment</string>
```

◦ ◦

```
public void beginFuturePayment(View view) {
    Intent serviceConfig = new Intent(this, PayPalService.class);
    serviceConfig.putExtra(PayPalService.EXTRA_PAYPAL_CONFIGURATION, config);
    startService(serviceConfig);
}
```



```

Intent intent = new Intent(this, PayPalFuturePaymentActivity.class);
intent.putExtra(PayPalService.EXTRA_PAYPAL_CONFIGURATION, config);
startActivityForResult(intent, 0);
}

```

- authCode metadataId **2**◦

```

@Override
protected void onActivityResult (int requestCode, int resultCode, Intent data){
    if (resultCode == Activity.RESULT_OK){
        PayPalAuthorization auth =
data.getParcelableExtra(PayPalFuturePaymentActivity.EXTRA_RESULT_AUTHORIZATION);
        if (auth != null){
            try{
                //prepare params to be sent to server
                String authCode = auth.getAuthorizationCode();
                String metadataId = PayPalConfiguration.getClientMetadataId(this);
                String [] params = {authCode, metadataId};

                //process async server request for token + payment
                ServerRequest req = new ServerRequest();
                req.execute(params);

            } catch (JSONException e) {
                Log.e("FPSample", "JSON Exception: ", e);
            }
        }
    } else if (resultCode == Activity.RESULT_CANCELED) {
        Log.i("FPSample", "User canceled.");
    } else if (resultCode == PayPalFuturePaymentActivity.RESULT_EXTRAS_INVALID) {
        Log.i("FPSample", "Invalid configuration");
    }
}
}

```

- onDestroy() ◦

```

@Override
public void onDestroy(){
    stopService(new Intent(this, PayPalService.class));
    super.onDestroy();
}

```

## Android2

[PayPal Developer Github](#) Android +◦

PayPalPayPal SDKID◦

- **URI** `http://10.0.2.2:3000/fpstore localhost/fpstore`◦
- **JSONID**◦
- ◦ **200/201**◦ ◦
- `onPostExecute(...)`◦ ◦

```

public class ServerRequest extends AsyncTask<String, Void, String> {
    protected String doInBackground(String[] params){
        HttpURLConnection connection = null;
        try{
            //set connection to connect to /fpstore on localhost
            URL u = new URL("http://10.0.2.2:3000/fpstore");
            connection = (HttpURLConnection) u.openConnection();
            connection.setRequestMethod("POST");

            //set configuration details
            connection.setRequestProperty("Content-Type", "application/json");
            connection.setRequestProperty("Accept", "application/json");
            connection.setAllowUserInteraction(false);
            connection.setConnectTimeout(10000);
            connection.setReadTimeout(10000);

            //set server post data needed for obtaining access token
            String json = "{\"code\": \"" + params[0] + "\", \"metadataId\": \"" + params[1] +
            "\"}";

            Log.i("JSON string", json);

            //set content length and config details
            connection.setRequestProperty("Content-length", json.getBytes().length + "");
            connection.setDoInput(true);
            connection.setDoOutput(true);
            connection.setUseCaches(false);

            //send json as request body
            OutputStream outputStream = connection.getOutputStream();
            outputStream.write(json.getBytes("UTF-8"));
            outputStream.close();

            //connect to server
            connection.connect();

            //look for 200/201 status code for received data from server
            int status = connection.getResponseCode();
            switch (status){
                case 200:
                case 201:
                    //read in results sent from the server
                    BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(connection.getInputStream()));
                    StringBuilder sb = new StringBuilder();
                    String line;
                    while ((line = bufferedReader.readLine()) != null){
                        sb.append(line + "\n");
                    }
                    bufferedReader.close();

                    //return received string
                    return sb.toString();
                }

            } catch (MalformedURLException ex) {
                Log.e("HTTP Client Error", ex.toString());
            } catch (IOException ex) {
                Log.e("HTTP Client Error", ex.toString());
            } catch (Exception ex) {
                Log.e("HTTP Client Error", ex.toString());
            } finally {

```

```

        if (connection != null) {
            try{
                connection.disconnect();
            } catch (Exception ex) {
                Log.e("HTTP Client Error", ex.toString());
            }
        }
    }
    return null;
}

protected void onPostExecute(String message) {
    //log values sent from the server - processed payment
    Log.i("HTTP Client", "Received Return: " + message);
}
}

```

## Android3

### PayPal Developer Github Android +.

2/fpstoreID. ◦

◦

```

var bodyParser = require('body-parser'),
    http = require('http'),
    paypal = require('paypal-rest-sdk'),
    app = require('express')();

var client_id = 'YOUR APPLICATION CLIENT ID';
var secret = 'YOUR APPLICATION SECRET';

paypal.configure({
  'mode': 'sandbox',
  'client_id': client_id,
  'client_secret': secret
});

app.use(bodyParser.urlencoded({ extended: false }))
app.use(bodyParser.json());

```

### ExpressAndroid/fpstorePOST. ◦

- POSTID. ◦
- generateToken() ◦ ◦
- payment.create(...) ◦ ◦

```

app.post('/fpstore', function(req, res) {
  var code = {'authorization_code': req.body.code};
  var metadata_id = req.body.metadataId;

  //generate token from provided code
  paypal.generateToken(code, function (error, refresh_token) {
    if (error) {

```

```

        console.log(error);
        console.log(error.response);
    } else {
        //create future payments config
        var fp_config = {'client_metadata_id': metadata_id, 'refresh_token':
refresh_token};

        //payment details
        var payment_config = {
            "intent": "sale",
            "payer": {
                "payment_method": "paypal"
            },
            "transactions": [{
                "amount": {
                    "currency": "USD",
                    "total": "3.50"
                },
                "description": "Mesozoic era monster toy"
            }]
        };

        //process future payment
        paypal.payment.create(payment_config, fp_config, function (error, payment) {
            if (error) {
                console.log(error.response);
                throw error;
            } else {
                console.log("Create Payment Response");
                console.log(payment);

                //send payment object back to mobile
                res.send(JSON.stringify(payment));
            }
        });
    }
});
});
});

```

3000。

```

//create server
http.createServer(app).listen(3000, function () {
    console.log('Server started: Listening on port 3000');
});

```

<https://riptutorial.com/zh-CN/paypal/topic/4537/-->

## 5:

	Express
bodyParser	JSONbody-parser
clientId	IDOAuth 2
HTTP	http
	PayPalSDK
	OAuth 2
webhookId	webhookID
webhookUpdate	webhookJSON

PayPal webhook。

## Examples

### ngrokExpressSandbox Webhooks

webhookngrokNode HTTPlocalhostInternet。 NodewebhookwebhookHTTP POST。

1. webhooksPOSTPayPallocalhost。
2. ngroklocalhostInternetPayPal。
3. webhook2ngrok URI。

### Webhooks

- ngrokURLwebhook。

```
var bodyParser = require('body-parser'),
    http = require('http'),
    app = require('express')();

app.use(bodyParser.json());

app.post('/', function(req, res){
  console.log(JSON.stringify(req.body));
});

//create server
http.createServer(app).listen(3001, function () {
  console.log('Server started: Listening on port 3001');
});
```

Express。 POSTPOST。 。

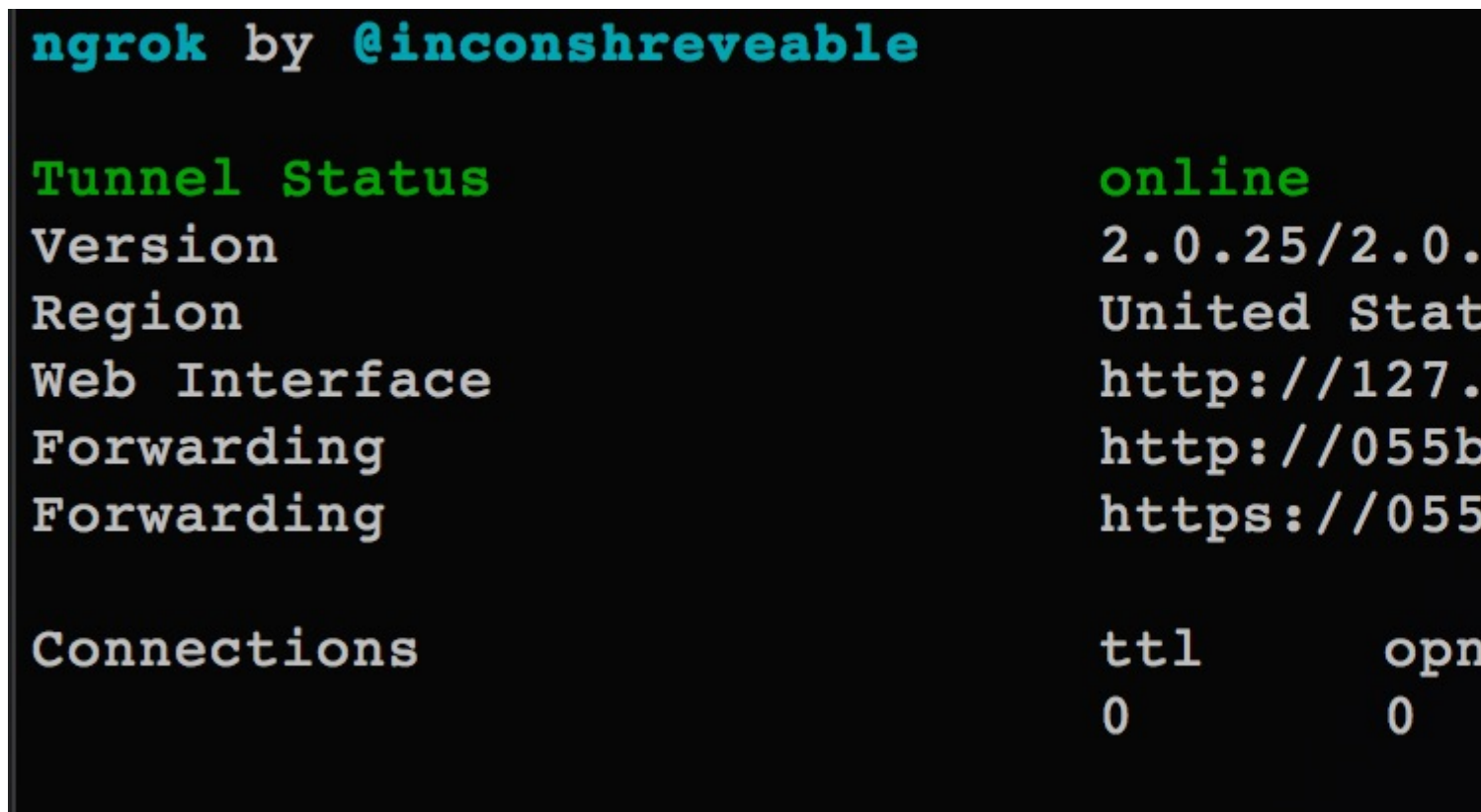
HTTPlocalhost3001。 。

## ngrokInternet

localhost3001ngrok。 。

```
ngrok http 3001
```

3001localhost



PayPal webhooklocalhosthttp(s)://055b3480.ngrok.io。 。

webhooks。 webhook。 。

PayPal Node SDKID /PayPal。 。

```
var paypal = require('paypal-rest-sdk');

var clientId = 'YOUR APPLICATION CLIENT ID';
var secret = 'YOUR APPLICATION SECRET';

paypal.configure({
  'mode': 'sandbox', //sandbox or live
  'client_id': clientId,
  'client_secret': secret
});
```

webhooksJSON◦ webhookswebhookurl event\_types◦

urlngrokURL◦

<https://developer.paypal.com/docs/integration/direct/rest-webhooks-overview/#event-type-support>

webhookswebhooks notification.webhook.create◦ PayPallocalhost◦

```
var webhooks = {
  "url": "https://436e4d13.ngrok.io",
  "event_types": [{
    "name": "PAYMENT.SALE.COMPLETED"
  }, {
    "name": "PAYMENT.SALE.DENIED"
  }
]};

paypal.notification.webhook.create(webhooks, function (err, webhook) {
  if (err) {
    console.log(err.response);
    throw error;
  } else {
    console.log("Create webhook Response");
    console.log(webhook);
  }
});
```

## PayPalPOSTPayPal

```
{
  "id": "WH-9FE9644311463722U-6TR22899JY792883B",
  "create_time": "2016-04-20T16:51:12Z",
  "resource_type": "sale",
  "event_type": "PAYMENT.SALE.COMPLETED",
  "summary": "Payment completed for $ 7.47 USD",
  "resource": {
    "id": "18169707V5310210W",
    "state": "completed",
    "amount": {
      "total": "7.47",
      "currency": "USD",
      "details": {
        "subtotal": "7.47"
      }
    },
    "payment_mode": "INSTANT_TRANSFER",
    "protection_eligibility": "ELIGIBLE",
    "protection_eligibility_type": "ITEM_NOT_RECEIVED_ELIGIBLE,UNAUTHORIZED_PAYMENT_ELIGIBLE",
    "transaction_fee": {
      "value": "0.52",
      "currency": "USD"
    },
    "invoice_number": "",
    "custom": ""
  }
}
```

```

"parent_payment": "PAY-809936371M327284GK4L3FHA",
"create_time": "2016-04-20T16:47:36Z",
"update_time": "2016-04-20T16:50:07Z",
"links": [
  {
    "href": "https://api.sandbox.paypal.com/v1/payments/sale/18169707V5310210W",
    "rel": "self",
    "method": "GET"
  },
  {
    "href":
"https://api.sandbox.paypal.com/v1/payments/sale/18169707V5310210W/refund",
    "rel": "refund",
    "method": "POST"
  },
  {
    "href": "https://api.sandbox.paypal.com/v1/payments/payment/PAY-
809936371M327284GK4L3FHA",
    "rel": "parent_payment",
    "method": "GET"
  }
]
},
"links": [
  {
    "href": "https://api.sandbox.paypal.com/v1/notifications/webhooks-events/WH-
9FE9644311463722U-6TR22899JY792883B",
    "rel": "self",
    "method": "GET"
  },
  {
    "href": "https://api.sandbox.paypal.com/v1/notifications/webhooks-events/WH-
9FE9644311463722U-6TR22899JY792883B/resend",
    "rel": "resend",
    "method": "POST"
  }
]
}

```

## URLWebhook

webhookURL。webhookPayPalID。

PayPal SDK。

```

var paypal = require('paypal-rest-sdk');

var clientId = 'YOUR APPLICATION CLIENT ID';
var secret = 'YOUR APPLICATION SECRET';

paypal.configure({
  'mode': 'sandbox', //sandbox or live
  'client_id': clientId,
  'client_secret': secret
});

```

JSONwebhook。webhookIDwebhookId。webhookUpdate replacepath/urlURLvalue。



```
var webhookId = "YOUR WEBHOOK ID";
var webhookUpdate = [{
  "op": "replace",
  "path": "/url",
  "value": "https://64fb54a2.ngrok.io"
}];
```

notification.webhook.replace(...) webhookIdwebhookUpdate ◦

```
paypal.notification.webhook.replacewebhookIdwebhookUpdatefunctionerrres{iferr{console.logerr;
throw err;} else {console.logJSON.stringifyres;}}
```

## PayPal◦

```
{
  "id": "4U496984902512511",
  "url": "https://64fb54a2.ngrok.io",
  "event_types": [{
    "name": "PAYMENT.SALE.DENIED",
    "description": "A sale payment was denied"
  }],
  "links": [{
    "href": "https://api.sandbox.paypal.com/v1/notifications/webhooks/4U496984902512511",
    "rel": "self",
    "method": "GET"
  }, {
    "href": "https://api.sandbox.paypal.com/v1/notifications/webhooks/4U496984902512511",
    "rel": "update",
    "method": "PATCH"
  }, {
    "href": "https://api.sandbox.paypal.com/v1/notifications/webhooks/4U496984902512511",
    "rel": "delete",
    "method": "DELETE"
  }],
  "httpStatusCode": 200
}
```

<https://riptutorial.com/zh-CN/paypal/topic/575/>

# 6: PayPal

clientId	PayPalDOAuth 2
	PayPalHATEOAS
paymentId	PayPalID
payerId	PayPalID
PayPal Node SDK	
payReq	JSON
REQ	
PayPalOAuth 2	

PayPal SDKPayPal. ◦

## Examples

### Node Express

PayPal Node SDKExpressPayPal. JSON. ◦

PayPal

1. JSONPayPal. PayPal. ◦
2. PayPal. PayPal. ◦
3. ◦

NodeNPMPayPal Node SDK

```
npm install paypal-rest-sdk
```

◦

```
var http = require('http'),
    paypal = require('paypal-rest-sdk'),
    bodyParser = require('body-parser'),
    app = require('express')();

var client_id = 'YOUR APPLICATION CLIENT ID';
var secret = 'YOUR APPLICATION SECRET';
```

```
//allow parsing of JSON bodies
app.use(bodyParser.json());

//configure for sandbox environment
paypal.configure({
  'mode': 'sandbox', //sandbox or live
  'client_id': client_id,
  'client_secret': secret
});
```

1. HTTP。
2. PayPalSDK。
3. bodyParserJSON。
4. Express。

ID。 bodyParserJSON。

PayPal。

```
app.get('/create', function(req, res){
  //build PayPal payment request
  var payReq = JSON.stringify({
    'intent':'sale',
    'redirect_urls':{
      'return_url':'http://localhost:3000/process',
      'cancel_url':'http://localhost:3000/cancel'
    },
    'payer':{
      'payment_method':'paypal'
    },
    'transactions':[{
      'amount':{
        'total':'7.47',
        'currency':'USD'
      },
      'description':'This is the payment transaction description.'
    }]
  });

  paypal.payment.create(payReq, function(error, payment){
    if(error){
      console.error(error);
    } else {
      //capture HATEOAS links
      var links = {};
      payment.links.forEach(function(linkObj){
        links[linkObj.rel] = {
          'href': linkObj.href,
          'method': linkObj.method
        };
      });

      //if redirect url present, redirect user
      if (links.hasOwnProperty('approval_url')){
        res.redirect(links['approval_url'].href);
      } else {
        console.error('no redirect URI present');
      }
    }
  });
});
```

```

    }
  });
});

```

JSONPayPal ◦ intentsale URL/payment\_methodpaypal ◦

payment.create(...) payReq ◦ PayPal ◦ HATEOASURLURLapproval\_url ◦

HATEOAS ◦ approval\_url ◦

PayPal ◦ createPayment(...)return\_url ◦

◦

```

app.get('/process', function(req, res){
  var paymentId = req.query.paymentId;
  var payerId = { 'payer_id': req.query.PayerID };

  paypal.payment.execute(paymentId, payerId, function(error, payment){
    if(error){
      console.error(error);
    } else {
      if (payment.state == 'approved'){
        res.send('payment completed successfully');
      } else {
        res.send('payment not successful');
      }
    }
  });
});

```

paymentId PayerIDtoken ◦ ◦

PayerID ◦ payment.execute(...) ◦

payment.stateapprovedpayment.state ◦ ◦

◦

```

//create server
http.createServer(app).listen(3000, function () {
  console.log('Server started: Listening on port 3000');
});

```

http://localhost:3000/create ◦

PayPal <https://riptutorial.com/zh-CN/paypal/topic/449/paypal>

# 7:

card_data	JSON
	PayPalJSON
CLIENT_ID	PayPalIDOAuth 2
	PayPal Node SDK
	PayPalOAuth 2
UUID	node-uuid

PayPal SDK。

## Examples

### NPMPayPal

```
npm install paypal-rest-sdk
```

### SDK

```
var paypal = require('paypal-rest-sdk');  
  
var client_id = 'YOUR CLIENT ID';  
var secret = 'YOUR SECRET';  
  
paypal.configure({  
  'mode': 'sandbox', //sandbox or live  
  'client_id': client_id,  
  'client_secret': secret  
});
```

SDKID。。

JSON。

```
var card_data = {  
  "intent": "sale",  
  "payer": {  
    "payment_method": "credit_card",  
    "funding_instruments": [{  
      "credit_card": {  
        "type": "visa",  
        "number": "4417119669820331",  
        "expire_month": "11",  
        "expire_year": "2018",
```

```

    "cvv2": "874",
    "first_name": "Joe",
    "last_name": "Shopper",
    "billing_address": {
      "line1": "52 N Main ST",
      "city": "Johnstown",
      "state": "OH",
      "postal_code": "43210",
      "country_code": "US" }}}},
"transactions": [{
  "amount": {
    "total": "7.47",
    "currency": "USD",
    "details": {
      "subtotal": "7.41",
      "tax": "0.03",
      "shipping": "0.03"}},
  "description": "This is the payment transaction description."
}]];

```

sale intent credit\_card payment\_method ◦ funding\_instruments funding\_instrument transactions transactions ◦ ◦

payment.create(...) card\_data ◦

```

paypal.payment.create(card_data, function(error, payment){
  if(error){
    console.error(error);
  } else {
    console.log(payment);
  }
});

```

```

{
  "id": "PAY-9BS08892W3794812YK4HKFQY",
  "create_time": "2016-04-13T19:49:23Z",
  "update_time": "2016-04-13T19:50:07Z",
  "state": "approved",
  "intent": "sale",
  "payer": {
    "payment_method": "credit_card",
    "funding_instruments": [
      {
        "credit_card": {
          "type": "visa",
          "number": "xxxxxxxxxxxx0331",
          "expire_month": "11",
          "expire_year": "2018",
          "first_name": "Joe",
          "last_name": "Shopper",
          "billing_address": {
            "line1": "52 N Main ST",
            "city": "Johnstown",
            "state": "OH",
            "postal_code": "43210",
            "country_code": "US"
          }
        }
      }
    ]
  }
}

```

```

    }
  ]
},
"transactions": [
  {
    "amount": {
      "total": "7.47",
      "currency": "USD",
      "details": {
        "subtotal": "7.41",
        "tax": "0.03",
        "shipping": "0.03"
      }
    },
    "description": "This is the payment transaction description.",
    "related_resources": [
      {
        "sale": {
          "id": "0LB81696PP288253D",
          "create_time": "2016-04-13T19:49:23Z",
          "update_time": "2016-04-13T19:50:07Z",
          "amount": {
            "total": "7.47",
            "currency": "USD"
          },
          "state": "completed",
          "parent_payment": "PAY-9BS08892W3794812YK4HKFQY",
          "links": [
            {
              "href":
"https://api.sandbox.paypal.com/v1/payments/sale/0LB81696PP288253D",
              "rel": "self",
              "method": "GET"
            },
            {
              "href":
"https://api.sandbox.paypal.com/v1/payments/sale/0LB81696PP288253D/refund",
              "rel": "refund",
              "method": "POST"
            },
            {
              "href": "https://api.sandbox.paypal.com/v1/payments/payment/PAY-
9BS08892W3794812YK4HKFQY",
              "rel": "parent_payment",
              "method": "GET"
            }
          ],
          "fmf_details": {
            },
            "processor_response": {
              "avs_code": "X",
              "cvv_code": "M"
            }
          }
        }
      ]
    },
    "links": [
      {

```

```

    "href": "https://api.sandbox.paypal.com/v1/payments/payment/PAY-9BS08892W3794812YK4HKFY",
    "rel": "self",
    "method": "GET"
  }
],
"httpStatusCode": 201
}

```

approved state ◦ links [HATEOAS](#) ◦ self [GET](#) ◦

## PayPal ◦

◦ IDPCI ◦

◦

```

var paypal = require('paypal-rest-sdk'),
    uuid = require('node-uuid');

var client_id = 'YOUR CLIENT ID';
var secret = 'YOUR SECRET';

paypal.configure({
  'mode': 'sandbox', //sandbox or live
  'client_id': client_id,
  'client_secret': secret
});

```

node-uuid [UUID](#) ◦

```
npm install node-uuid
```

[PayPalJSON](#) ◦ [node-uuidID](#) ◦ [payer\\_id](#) ◦

```

var create_card_details = {
  "type": "visa",
  "number": "4417119669820331",
  "expire_month": "11",
  "expire_year": "2018",
  "first_name": "John",
  "last_name": "Doe",
  "payer_id": uuid.v4()
};

```

◦ [credit\\_card.create\(...\)](#) [credit\\_card\\_details](#) ◦ ◦ [payer\\_idID](#) ◦

```

paypal.credit_card.create(create_card_details, function(error, credit_card){
  if(error){
    console.error(error);
  } else {
    var card_data = {
      "intent": "sale",
      "payer": {

```



```

        "payment_method": "credit_card",
        "funding_instruments": [{
            "credit_card_token": {
                "credit_card_id": credit_card.id,
                "payer_id": credit_card.payer_id
            }
        }]
    },
    "transactions": [{
        "amount": {
            "total": "7.47",
            "currency": "USD",
            "details": {
                "subtotal": "7.41",
                "tax": "0.03",
                "shipping": "0.03"
            }
        },
        "description": "This is the payment transaction description."
    }]
};

paypal.payment.create(card_data, function(error, payment){
    if(error){
        console.error(error);
    } else {
        console.log(JSON.stringify(payment));
    }
});
});

```

◦ card\_data payer\_funding\_instruments ◦ **IDID**

```

"credit_card_token": {
    "credit_card_id": credit_card.id,
    "payer_id": credit_card.payer_id
}

```

◦

<https://riptutorial.com/zh-CN/paypal/topic/444/-->

---

S. No		Contributors
1	PayPal	<a href="#">Community</a> , <a href="#">Jonathan LeBlanc</a> , <a href="#">Nathan Arthur</a>
2	/	<a href="#">Jonathan LeBlanc</a>
3	PayPal /	<a href="#">Jonathan LeBlanc</a>
4		<a href="#">Jonathan LeBlanc</a>
5		<a href="#">Jonathan LeBlanc</a>
6	PayPal	<a href="#">Jonathan LeBlanc</a>
7		<a href="#">Jonathan LeBlanc</a>