

 無料電子ブック

学習

PayPal

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

#paypal

.....	1
<b>1: PayPal</b> .....	<b>2</b>
.....	2
.....	2
Examples .....	2
ID / .....	2
.....	4
<b>2: PayPal</b> .....	<b>5</b>
.....	5
.....	5
Examples .....	5
.....	5
<b>3: Webhooks</b> .....	<b>9</b>
.....	9
.....	9
Examples .....	9
ngrokExpressWebhook .....	9
URLWebhook .....	13
<b>4:</b> .....	<b>15</b>
.....	15
.....	15
Examples .....	15
.....	15
.....	18
<b>5: PayPal /</b> .....	<b>21</b>
.....	21
.....	21
Examples .....	21
AndroidPayPal / .....	21
<b>6:</b> .....	<b>25</b>
.....	25

Examples.....	25
Android1.....	25
Android2.....	26
Android3.....	28
<b>7: /.....</b>	<b>31</b>
.....	31
.....	31
Examples.....	32
2.....	32
1.....	34
.....	<b>37</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をいめる

これらのガイドでは、アプリケーション、アカウントなどのアカウントをユーザにします。このガイドには、PayPal APIのになものがすべてまれています。

## バージョン

バージョン	
1.0.0	2016-04-11

## Examples

アプリケーションのとクライアントID /の

PayPal APIをしてをするには、クライアントIDとをするためのアプリケーションをするがあります。

にすように、 <https://developer.paypal.com/developer/applications/>にアクセスしてサインインし、[アプリケーションの]をクリックします。

## 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
<a href="#">My test app</a>
<a href="#">My test app 1</a>
<a href="#">MyLiveApp</a>

に、アプリケーションをし、するサンドボックステストアカウントをしますしいアカウントのはデフォルトのまま。[アプリケーションの]をクリックします。

## 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

アプリケーションがされると、サンドボックスとライブクライアントのIDとシークレットがされ  
ます。これはのようになります。

## 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

これらのクレデンシャルは、アプリケーションをしてリクエストをうためにPayPal APIにリクエ  
ストをうときにするクレデンシャルです。

サンドボックスユーザーテストアカウントの

サンドボックスでのPayPalのをテストするときは、いフローをするためにするサンドボックスユーザーアカウントをするがあります。

<https://developer.paypal.com/developer/accounts/>にアクセスし、PayPalアカウントをしてログインし、のように[アカウントの]をクリックします。

## 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

のメール、アカウント、い、などをむしいテストユーザーのアカウントのをし、したページにある「アカウントの」をクリックします。これにより、しいアカウントがされます。

このしいユーザーのアカウントのをするには、アカウントページのエントリをし、[プロフィール]をクリックします。

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

そのプロフィールがみまれたら、[]タブをクリックすると、サンドボックスにするのクレジットカードにされるクレジットカードなど、そのアカウントのいがされます。

サンドボックスAPIエンドポイントをするは、サンドボックステストアカウントをしてログインし、テストをうがあります。これは、ライブアカウントがしないためです。

オンラインでPayPalをいめるをむ <https://riptutorial.com/ja/paypal/topic/406/paypal>をいめる

## 2: PayPalでのい

### パラメーター

パラメータ	
クライアントID	PayPalアプリケーションクライアントID OAuth 2の
リンク	PayPalからのすべてのHATEOASリターンのなオブジェクト
paymentId	いをするためにPayPalからされたいのID
payerId	いをするためにPayPalからされたのID
ペイパル	PayPalノードSDKリファレンス
payReq	トランザクションのをむJSONオブジェクト
req	サーバーからのオブジェクト
レス	サーバーからのオブジェクト
の	あなたのPayPalアプリケーションのOAuth 2の

これらのサンプルでは、PayPal SDKをしてPayPalでいをするについてしています。これらは、このいオプションをするためのステップのプロセスをするなリクエストサンプルです。

## Examples

### ノード・エクスプレス・サーバーの

このでは、PayPalノードSDKをして、PayPalでいをするをするExpressサーバーをします。にするため、いにはなJSONをします。

PayPalいをするをするには、の3つのなにいます。

1. PayPalでするのをむJSONオブジェクトをします。その、PayPalにして、いをするためにユーザーをリダイレクトするリンクをします。
2. に、PayPalにリダイレクトしていをします。がむと、PayPalはユーザーをアプリケーションにリダイレクトします。
3. アプリにしたら、ユーザーにわっていをします。

これをなNodeアプリケーションとしてすると、まずNPMからPayPal Node SDKをします。



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

に、アプリのとパッケージをします。

```
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
});
```

このアプリには4つのがです

1. たちのサーバーのHTTPパッケージ。
2. PayPalノードSDKパッケージ。
3. JSONエンコードされたボディをするbodyParserパッケージ。
4. たちのサーバーのExpressフレームワーク。

のは、[アプリケーションの](#)にされたクライアントIDとシークレットのをします。bodyParserをしてJSONエンコードされたボディをし、アプリケーションのをしてアプリケーションをし、のをしますのライブまたはテストのサンドボックス。

PayPalでいリクエストをするルートをししましょう。

```
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');
    }
  }
});
});

```

にうのは、いをするためにPayPalにするのあるがまれているいJSONオブジェクトをすることで。たちは、intentするsaleにし、PayPalは、らがいキャンセルにユーザをするがありますリダイレクトURLをし、payment\_methodのpaypal々はPayPalいをいますことをらせるためには、そのためのトランザクションをしますする。

に、payReqオブジェクトをしてpayReq payment.create(...)びします。これにより、PayPalにがされます。それがってしたら、returnオブジェクトのHATEOASリンクをループして、ユーザーをリダイレクトするがあるURLをします。このURLはapproval\_urlにされます。

HATEOASリンクのフォーマットは、するとれやすいコードをするがあります。そのため、されたすべてのリンクをループして、よりいオブジェクトにれることで、のをします。そのオブジェクトにapproval\_urlがつかった、ユーザーをリダイレクトします。

こので、ユーザーはいをするためにPayPalにリダイレクトされます。それらがされると、createPayment(...)でしたreturn\_urlリダイレクトされます。

いをさせるためには、をするルートをするがあります。

```

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');
      }
    }
  });
});

```

```
    }  
  });  
});
```

ユーザーがあなたのアプリにアクセスすると、`paymentId`、`PayerID`、および`token` 3つのクエリパラメータも`token`。々はの2つにするがあります。

パラメータをし、ステップののために`PayerID`をなオブジェクトにします。に、`payment.execute(...)`をびし、2つのパラメータをしていをしします。

そのがわれると、`payment.state`が`approved`されているかどうかをして、いがにしたかどうかをしします。その、されたオブジェクトからなものをすることができます。

たちののステップは、サーバーをし、したルートにするトラフィックをちけることです。

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

サーバーがされると、`http://localhost:3000/create`にくと、いがされます。

オンラインでPayPalでのいをもむ <https://riptutorial.com/ja/paypal/topic/449/paypalでのい>

## 3: Webhooks

### パラメーター

パラメータ	
アプリ	Expressアプリケーションのリファレンス
bodyParser	JSONエンコードされたボディをするbody-parserパッケージリファレンス
クライアントID	アプリケーションクライアントID OAuth 2の
http	サーバーをするためのhttpパッケージ
ペイパル	PayPalノードSDKオブジェクト
の	アプリケーションのOAuth 2の
webhookId	するウェブフックのID
webhookUpdate	されるwebhookのをむJSONオブジェクト

これらのサンプルは、PayPalウェブフックをしてアプリケーションといのイベントをするのをカバーしています。

## Examples

### ngrokとExpress ノードをしたサンドボックスWebhookのテスト

このでは、サンドボックスでwebhookをテストし、[ngrok](#)をして、localhostでされているNode HTTPリスナーのトンネルをインターネットにするをていきます。このでは、ノードをしていイベントいがわれているなどのウェブフックをし、WebhookイベントからのHTTP POSTメッセージをするようにサーバーをします。

これをするためにここにうべきいくつかのステップがあります

1. WebhooksからのPOSTトラフィックをくためのシンプルなサーバーをします。これはPayPalからのであり、localhostでのリスニングをします。
2. その、ngrokをしてローカルホストからインターネットへのトンネルをし、PayPalがをできるようにします。
3. に、されたについてたちがしたいwebhookイベントにアプリケーションをし、ステップ2のpublic ngrok URIをします。

## Webhooks リスナーの

まず、リスナーをするがあります。リスナーをするは、webhooksをまたはするにngrokのライブURLがなためです。

```
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');
});
```

たちのリスナーはExpressをつたなルートです。ってくるPOSTトラフィックをちけ、POSTをコンソールにきします。リスナーとにたちがきなことをするためにこれをうことができます。

にHTTPサーバーをするときは、localhostポート3001でするようにします。このスクリプトをすぐして、トラフィックのをします。

### ngrokをしてリスナーをインターネットにする

リスナーをlocalhost3001にして、のはそのスクリプトをインターネットにして、ngrokのであるトラフィックをできるようにすることです。

ターミナルウィンドウからのコマンドをします。

```
ngrok http 3001
```

これにより、ポート3001でローカルホストのライブトンネルをするプロセスがされ、されることがされます。

## ngrok by @inconshreveable

```
Tunnel Status      online
Version            2.0.25/2.0.
Region             United Stat
Web Interface      http://127.
Forwarding         http://055b
Forwarding         https://055

Connections        ttl      opn
                   0        0
```

たちがることができるように、localhostのリスナーにPayPal webhookをすためにできるライブアドレスは、 `http(s)://055b3480.ngrok.io` です。リスナーをするには、それだけです。

をする

たちのステップは、たちのアプリケーションのウェブフックをすることです。これは、たちのアプリでいやいしなどのイベントがしたときにをします。これらのwebhookは、アプリケーションにバインドするためにするだけでみ、するたびにするはありません。

に、PayPalノードSDKのをし、クライアントID /シークレットでアプリケーションをし、にサンドボックスのをすることで、PayPalをセットアップしました。

```
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
});
```

に、webhooksのJSONをします。webhooksは、2つの、すべてのwebhookイベントをするurl、およびしたいevent\_typesが含まれています。

このサンプルの、urlはGoogleのライブURLにされており、おちしているイベントはいがまたはされたです。

なイベントのなりリストについては、 <https://developer.paypal.com/docs/integration/direct/rest-webhooks-overview/#event-type-support>をしてください。

に、 `webhooks` オブジェクトをびして `webhooks` をします `notification.webhook.create`。した、 PayPalはしたエンドポイント `localhost` でにをします。

```
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);
  }
});
```

これらのアプリケーションをしていをする、いにするが、したエンドポイントにされます。

PayPalがとしてするPOSTのは、PayPalのいがしたにされたのようなものです。

```
{
  "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"
  }
]
}

```

## しいURLをしたWebhookのノードサンプル

このサンプルでは、のwebhookURLをPOSTするをするをします。これをするには、にウェブフックをしたときにPayPalからされたIDをとっているがあります。

まず、PayPal SDKをしてをしますのサンドボックス。

```

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
});

```

に、JSONとウェブフックのをします。ためにあなたのウェブフックのためのIDをりてwebhookId。に、webhookUpdateでreplaceのをし、/urlへのpathをしてそのリソースのをし、しいURLをして



そのvalueにきえvalue。

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

に、webhookIdとwebhookUpdateをして、notification.webhook.replace(...)をwebhookUpdateます。

JSON.stringify(JSON.stringify(res));

すべてしたは、のようなオブジェクトをPayPalからすがあります。このサンプルのは、しくがされたにされます。

```
{
  "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
}
```

オンラインでWebhooksをむ <https://riptutorial.com/ja/paypal/topic/575/webhooks>

## 4: クレジットカードノードの

### パラメーター

パラメータ	
card_data	トランザクションのをむJSONオブジェクト
クレジットカードの	ペイパルにされてポルトされるクレジットカードデータをむJSONオブジェクト
クライアントID	PayPalアプリケーションクライアントID OAuth 2の
ペイパルの	PayPalノードSDKリファレンスのあなたのPayPalアプリケーションのOAuth 2の
uuid	node-uuidパッケージへの

このサンプルでは、PayPal SDKをしたなクレジットカードのクレジットをユーザーにしています。

## Examples

### ノードサンプル

NPMからPayPalノードモジュールをインストールすることからめる

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

アプリケーションファイルで、SDKのをします

```
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
});
```

SDKのをし、[アプリケーションを](#)するにされたクライアントIDとシークレットのをします。これらのをしてアプリケーションをし、のライブまたはサンドボックスをします。

に、のいをむJSONオブジェクトをします。

```
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のintentをし、 payment\_methodをcredit\_cardます。に、 funding\_instrumentsあるクレジットカードのカードのと、 transactionsでされるをします。のトランザクションオブジェクトをここにできます。

に、 card\_data payment.create(...) リクエストをい、 card\_dataオブジェクトをしていをします。

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

トランザクションがしたは、のようなオブジェクトがされます。

```
{
  "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-9BS08892W3794812YK4HKFQY",
    "rel": "self",
    "method": "GET"
  }
],
"httpStatusCode": 201
}

```

このオブジェクトでは、`approved`の`state`でトランザクションがしたことがわかります。`links`オブジェクトには、されたばかりのアクションでされる次のステップをするのHATEOASリンクがあります。この、された`self`エンドポイントへのGETをうことで、いにするをりすことができます。

## アーチのクレジットカードノードでいをう

このでは、PayPalをしてクレジットカードをするをし、されたクレジットカードをして、ユーザーのクレジットカードをするをします。

をするは、のサーバーにクレジットカードをするがないためです。されたポールトIDをしていをするだけで、クレジットカードのにするくのPCIコンプライアンスにするはありません。

のサンプルとに、たちはをすることからめます。

```

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
});

```

のサンプルとの1つのいは、しいパッケージ`node-uuid`がであることです。このパッケージは、カ

ードをするときにのUUIDをするためにされます。あなたはそのパッケージをのでインストール  
することができます

```
npm install node-uuid
```

に、PayPalににされるクレジットカードJSONオブジェクトをします。これには、カードからの  
と、node-uuidをしてするのIDがまわていnode-uuid。こののpayer\_idは、アーチのカードでいをする  
るにされるため、のデータベースにするがあります。

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

に、クレジットカードをし、そのカードをしていをするがあります。クレジットカードをするた  
めに、たちはしたcredit\_card\_detailsオブジェクトをしてcredit\_card.create(...)をびします。す  
べてがうまくいけば、ポールトされたカードについてののをすオブジェクトをさなければなりませ  
ん。そのカードでのいのために、たちはすでにしているpayer\_idとポールトIDの2つのしかとしま  
せん。これはたちのデータベースのとしてもするがあります。

```
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));
        }
    });
}
});
```

クレジットカードのボーリングがしたのセクションでは、のクレジットカードの、カードのをしていをするだけです。 `card_data` オブジェクトのの、 `payer` している `funding_instruments` セクションです。クレジットカードをするわりに、ボールトIDとIDをむのオブジェクトをします。

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

これは、をするためにカードをするです。

オンラインでクレジットカードノードのをむ <https://riptutorial.com/ja/paypal/topic/444/クレジットカード-ノード-の>

## 5: モバイル PayPal / クレジットカード

### パラメーター

パラメータ	
ボタン	ボタン
	クライアントIDアプリケーションのとするサンドボックスまたはライブをしたPayPalオブジェクト
い	PayPalいの
paymentConfig	いとどの
serviceConfig	パラメータデータの

でのいにするサンプル

## Examples

### Android PayPal / クレジットカードをける

このチュートリアルでは、PayPalいまたはクレジットカードによるないをするために [PayPal Android SDK](#) をするをします。こののには、クリックするとユーザーがPayPalにされ、されたいがされ、ユーザーがアプリケーションにいていのをするなボタンがアプリケーションにあります。

このサンプルのなアプリケーションコードは、 [PayPal Developer Github](#) リポジトリにあります。

めましょう。

のは、 [SDKをしてプロジェクト](#) にすることです。 build.gradle へののをのようになります。

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

に、 MainActivity.java ファイルまたは PayPal ボタンのをするにし、たちがするクライアントIDとサンドボックスの config オブジェクトをします。

```
private static PayPalConfiguration config = new PayPalConfiguration()
```



```
.environment(PayPalConfiguration.ENVIRONMENT_SANDBOX)
.clientId("YOUR CLIENT ID");
```

ここで、`onCreate(...)`メソッドでボタンをします。これにより、クリックされたPayPalによるい  
がになります。

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

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

これで、そのボタンのをすることがあります。あなたの`res>レイアウト>メイン`のXMLファイルでは  
、ボタンのテキストと`onClick`ハンドラををするボタンのを`paypal_button` IDのボタンにすることが  
できます。

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

クリックすると、ボタンは`beginPayment(...)`メソッドをびします。に、ボタンのテキストを  
`strings.xml`ファイルにすると、のようになります。

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

ボタンをのにいたまま、いをするためにボタンクリックをするがあります。の`onCreate(...)`メソ  
ッドののにの`beginPayment(...)`メソッドをします。

```
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);
}
```

ここでは、まずクライアントIDとサンドボックスでした`config`をして、サービスインテント  
`serviceConfig` を`config`します。に、するオブジェクトをします。このでは、およびをしてい  
ます。なアプリケーションでは、これらのは、ユーザーがアプリケーションでしようとしている  
ものからするがあります。に、`paymentConfig`をし、にした`config`オブジェクトと`payment`オブジェ  
クトをして、アクティビティをします。

ここで、ユーザーはPayPalのログインといがされ、PayPalまたはクレジットカードでうかどうかをすることができますまたはカメラがなはcard.io。そのはのようになります。

したら、いやキャンセルのにPayPalがユーザーをアプリケーションにすができたハンドラをするがあります。そののためにonActivityResult(...)をオーバーライドonActivityResult(...)ましよう

```
@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(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(...)メソッドで、resultCodeれるresultCodeがRESULT\_OKユーザーい、RESULT\_CANCELEDユーザーキャンセルい、またはRESULT\_EXTRAS\_INVALIDにがありますかどうかをしています。なの、いからされたオブジェクトがされ、このサンプルではログにされます。されるものは、のようなものになります。

```
{
  "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"
}
```

responseオブジェクトをと、approved stateであることがわかります。これは、いがされたことをします。ここで、サーバーにそのオブジェクトをして、いがにしたことをするがあります。これらのについては、これらのドキュメントをしてください。

たちのステップは、たちの `onDestroy(...)` クリーンアップすること `onDestroy(...)`。

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

それがすべてです。ここでは、PayPalまたはクレジットカードでいをするなボタンをしました。このから、このサンプルをするためのものがいくつかあります。

- `beginPayment(...)` メソッドでユーザーの `beginPayment(...)` に `beginPayment(...)`。
- サーバーに `beginPayment(...)` をし、 `beginPayment(...)` が `beginPayment(...)` になったことをします。
- エラーをし、アプリのユーザーのキャンセルをキャンセルします。

オンラインでモバイル PayPal / クレジットカードをむ <https://riptutorial.com/ja/paypal/topic/608/モバイルpaypal--クレジットカード>

## 6: モバイルののいエンド ツー エンド アプリケーション

これは、ノードサーバーをしてAndroidデバイスからPayPalののいをするなをしています。

### Examples

Androidのステップ1レイアウト、およびハンドリングサーバーの

このアプリケーションAndroid + ノードサーバーのサンプルコードは、[PayPal DeveloperのGithubリポジトリにあります](#)。

アプリケーションのAndroidをするのは、レイアウトをし、ノードをするサーバーからされたをすることです。

まず、しいPayPalConfigurationオブジェクトをしてアプリケーションをします。

```
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"));
```

に、なボタンをonCreate(...)にしていのをいます。これはにアクションをトリガーすることであり、ユーザーののいをするためのプロセスとしてするがありますたとえば、サブスクリプションにしたなど。

```
@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.xmlで、ボタンののをけられたアクションとともにします。クリックするとbeginFuturePayment(...) コールされます。これは1にします。

```
<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で、ボタンののをします。

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

ここで、ボタンハンドラをして、ユーザがボタンをクリックしたときにのいをするびしをします。ここでは、こののにしたオブジェクトでいサービスをします。

```
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);
}
```

のいをうためのびしがされると、サーバーにするがあるがされます。なのい authCode と metadataId からこのをし、サーバーにをして、のいをさせます2でしくします。

```
@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");
    }
}
```

に、 onDestroy() をします。

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

## Androidのステップ2サーバーリクエスト

このアプリケーションAndroid +ノードサーバーのサンプルコードは、 [PayPal DeveloperのGithubリポジトリ](#)にあります。

このでPayPalのいのいボタンがクリックされました.PayPal SDKからのコードとメタデータIDがあり、のいをするために、これらのをサーバーにすがあります。

のバックグラウンドプロセスでは、いくつかのことをしています。

- たちは、たちのサーバのURIを `http://10.0.2.2:3000/fpstore` にしました。これはlocalhostでされているサーバの `/fpstore` エンドポイントにたっています。
- されるJSONオブジェクトがされ、コードとメタデータIDがまれます。
- がわられます。リクエストがした200/201の、サーバーからがされることが出来ます。たちはそのをみ、それをします。
- に、されたサーバーをする `onPostExecute(...)` メソッドがされています。このの、にログにされます。

```
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);
}
}
}

```

## Androidのステップ3 ノードサーバーでアクセストークンとプロセスのいをする

このアプリケーションAndroid + ノードサーバーのサンプルコードは、 [PayPal DeveloperのGithubリポジトリ](#)にあります。

ステップ2から、 /fpstore エンドポイントのサーバーにコードとメタデータIDをすがわれました。をし、のいをするためにトークンをするがあります。

まず、とオブジェクトをします。

```

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());
```

これで、Androidコードから`/fpstore`エンドポイントにされたPOSTリクエストをリッスンするExpressルートをしました。

私たちはこのルートでいくつかのことをしています

- POSTからコードとメタデータIDをします。
- に、`generateToken()`メソッドにコードオブジェクトをしてリクエストします。すると、いをするためにできるトークンがされます。
- に、オブジェクトがされるのいにしてされ、`payment.create(...)`へのがわれ、のいおよびいオブジェクトにとってされます。これにより、のいがされます。

```
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));
    }
    });
}
});
});
```

に、サーバーをしてポート3000でするようにします。

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

オンラインでモバイルののいエンドツーエンドアプリケーションをむ

<https://riptutorial.com/ja/paypal/topic/4537/モバイルののい-エンドツーエンドアプリケーション->

## 7: 払いの

### パラメーター

パラメータ	
billingAgreementAttributes	をすするオブジェクト
プラン	クエリからのプランID
billingPlanAttribs	をすするオブジェクト
billingPlanUpdateAttributes	プランをなにするためのオブジェクト
クライアントID	アプリケーションクライアントID OAuthキー
http	シンプルなサーバーをセットアップするためのhttpパッケージへの
isoDate	サブスクリプションのをすするためのISO
リンク	PayPalへのリダイレクトURLをすするためのHATEOASリンクオブジェクト
パラメータ	クエリパラメータ
ペイパルの	PayPal SDKへのアプリケーションのOAuthキー
トークン	PayPalのリダイレクトにされたトークンが、をします。

これらのは、PayPalをしていシステムをすするプロセスをしています。

サブスクリプションをすするプロセスはのとおりです。

1. をします。これは、サブスクリプションのをすするなモデルです。
2. プランをにします。
3. ユーザーのサブスクリプションをすするは、サブスクリプションするののあるのIDをして、をします。
4. したら、ユーザーをPayPalにリダイレクトして、をします。されると、ユーザはマーチャントのウェブサイトのリダイレクトされる。
5. に、をしてサブスクリプションをします。

## Examples

ステップ2をしてユーザのサブスクリプションをするノードサンプル

ユーザのサブスクリプションをするための2のステップは、のなについてをしてすることです。このでは、のすでにをしてしており、このではそののIDをすることをとしています。

ユーザーのをするためのをするは、の3つのにいます。これは、PayPalいのをいこさせるものです。

1. IDをしてプランをして、をします。
2. したら、ユーザをPayPalにリダイレクトしてPayPalでっている、をします。がむと、PayPalは、のプランでされているリダイレクトをして、ユーザーをサイトにリダイレクトします。
3. その、PayPalリダイレクトをしてってきたトークンをしてをします。

このでは、ExpressベースのHTTPサーバーをしてプロセスをしています。

このをするには、まずをうがあります。 PayPal SDK、JSONエンコードされたボディをうbody-parser、シンプルなサーバーのためのhttp、Expressフレームワークのためのexpress 4つのをします。に、クライアントIDとシークレットをアプリケーションのからし、SDKをサンドボックスにし、bodyParserをJSONのにします。

```
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());
```

このステップは、のをするルートをし、ユーザーをPayPalにリダイレクトしてサブスクリプションをすることです。このIDでのURLをロードするなど、クエリプランのIDがクエリパラメータとしてされるとしています。

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

このをしてをするがあります。

```
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');
    }
  }
});
});

```

まず、クエリからIDをし、をするをします。

のオブジェクトである `billingAgreementAttributes` は、サブスクリプションのでされています。これには、にする、プランIDへの、い、およびにながされます。

に、 `billingAgreement.create(...)` びしがわれ、した `billingAgreementAttributes` オブジェクトが `billingAgreementAttributes` れます。すべてしたは、しくされたサブスクリプションにするをむオブジェクトをにりすがあります。このオブジェクトには、このしくされたでることができるのス

トップをするのHATEOASリンクもまれています。ここになるのは、 `approval_url` です。

されたすべてのリンクをループして、それらをにできるオブジェクトにれます。 `approval_url` がこれらのリンクの1つである、ユーザーをそのリンクPayPalにリダイレクトします。

ここで、ユーザはPayPalでをし、プランにされているURLにリダイレクトされます。そのURLにえて、PayPalはクエリによってトークンもします。そのトークンは、サブスクリプションをまたはするためにするものです。

のルートでそのをしましょう。

```
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');
    }
  });
});
```

クエリからトークンをし、 `billingAgreement.execute` びして、そのトークンをしします。すべてしたは、そのユーザーにしてなサブスクリプションがされます。オブジェクトには、なにをするがまれています。

に、HTTPサーバをしてルートへのトラフィックをちけます。

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

## 1をしたサブスクリプションモデルのノードサンプル

ユーザーのサブスクリプションをするときは、まず、ユーザーがをしてサブスクライブしているプランをしてするがあります。サブスクリプションをするためのプロセスは、このトピックのでしくしています。

ここでは、 [PayPalノードSDK](#)をしします。のコマンドをして、NPMからできます。

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

.jsファイルでは、にSDKのをし、クライアントIDとシークレットをしてアプリケーションをした、SDKをサンドボックスにするなど、SDKをいます。

```

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
});

```

に、2つのJSONオブジェクトをすることがあります。 `billingPlanAttribs` オブジェクトには、ユーザーをサブスクライブすることができるプランのといのがまれています `billingPlanUpdateAttributes` オブジェクトには、プランをアクティブにすることができるようにするがまれています。

```

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` オブジェクトには、するがいくつかあります。

- **//** タイプ プランをするためのなビジュアル、およびプランのタイプ。
- **payment\_definitions** プランがどのようにし、されるべきかにする。フィールドのは [こちら](#)。

- **merchant\_preferences**、リダイレクトURL、およびサブスクリプションプランの。フィールドのは[こちら](#)。

これらのオブジェクトをして、をしてすることができます。

```
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(...)`をびして、したばかりの`billingPlanAttribs`オブジェクトを  
`billingPlanAttribs`ます。それがした、オブジェクトにはにするがまれます。このでは、をにする  
ためにプランIDをするだけです。

に、にしたプランIDと`billingPlanUpdateAttributes`オブジェクトをして`billingPlan.update(...)`をび  
します。それがした、のプランはで、すぐにできます。

このプランでユーザーまたはのユーザーのサブスクリプションをするには、の`billingPlan.id`をす  
るがあります。そのため、にできるにしてください。

2のサブスクリプション・ステップでは、したについてをし、それをしてユーザーのサブスクリプ  
ションのをするがあります。

オンラインで/[いのをむ https://riptutorial.com/ja/paypal/topic/467/-いの](https://riptutorial.com/ja/paypal/topic/467/-いの)

## クレジット

S. No		Contributors
1	PayPalをいめる	<a href="#">Community</a> , <a href="#">Jonathan LeBlanc</a> , <a href="#">Nathan Arthur</a>
2	PayPalでのい	<a href="#">Jonathan LeBlanc</a>
3	Webhooks	<a href="#">Jonathan LeBlanc</a>
4	クレジットカードノ ードの	<a href="#">Jonathan LeBlanc</a>
5	モバイルPayPal /ク レジットカード	<a href="#">Jonathan LeBlanc</a>
6	モバイルののいエン ドツーエンドアプリ ケーション	<a href="#">Jonathan LeBlanc</a>
7	/いの	<a href="#">Jonathan LeBlanc</a>