pagarme-python Documentation

Versão 1.0.1

Allisson Azevedo

24/12/2014

Sumário

CAPÍTULO 1

Conteúdo

1.1 Instalação

O pagarme-python necessita do python versão 2.6+:

```
pip install pagarme-python
```

1.1.1 Configuração inicial

Faça o import do PagarMe e sete sua api_key.

```
>>> from pagarme import PagarMe
>>> PagarMe.api_key = 'api_key'
```

1.2 Transações

Antes de continuar, não esqueça de checar a documentação do pagar.me.

Não esqueça de setar sua api_key.

1.2.1 Criando dados de Metadata

Você pode enviar dados adicionais para a futura transação usando o metadata.

```
>>> from pagarme.metadata import MetaData
>>> metadata = MetaData(order_id=1, user_id=1)
```

1.2.2 Criando dados do Customer

É interessante que você envie os dados do cliente ao realizar a transação.

Se você estiver com o antifraude ligado, esses dados são obrigatórios.

```
>>> from pagarme.customer import Customer
>>> customer = Customer(
...     name='John Appleseed',
...     document_number='92545278157',
```

```
...     email='jappleseed@apple.com',
...     address_street='Av. Brigadeiro Faria Lima',
...     address_neighborhood='Jardim Paulistano',
...     address_zipcode='01452000',
...     address_street_number='2941',
...     address_complementary='8° andar',
...     phone_ddd='11',
...     phone_number='30713261'
... )
```

1.2.3 Criando uma transação usando cartão de crédito

Para realizar transações com cartão, você deve capturar os dados do cartão ou usar um card_id.

```
>>> from pagarme.transaction import Transaction
>>> card_hash = 'card_hash'
>>> transaction = Transaction(
...     amount='10000',
...     card_hash=card_hash,
...     customer=customer,
...     metadata=metadata,
...     postback_url='http://requestb.in/18j98r31',
...     soft_descriptor='Pagamento 1'
... )
>>> transaction.charge()
>>> transaction.data
{u'date_updated': u'2014-12-23T20:05:02.000Z', u'ip': u'179.179.108.26', u'boleto_barcode': None, u'
```

Uma boa dica é usar a lib python-dateutil para converter os formatos de data.

```
>>> # pip install python-dateutil
>>> from dateutil.parser import parse
>>> parse(transaction.data['date_updated'])
datetime.datetime(2014, 12, 23, 20, 5, 2, tzinfo=tzutc())
```

Você pode criar uma transação passando o card_id no lugar do card_hash.

```
>>> transaction = Transaction(
...     amount='10000',
...     card_id='card_ci3xq3kyu0000yd16rihoplu6',
...     customer=customer,
...     metadata=metadata,
...     postback_url='http://requestb.in/18j98r31',
...     soft_descriptor='Pagamento 1'
... )
>>> transaction.charge()
>>> transaction.data
{u'date_updated': u'2014-12-23T20:10:03.000Z', u'ip': u'179.179.108.26', u'boleto_barcode': None, u'
```

Você pode optar por não passar o postback_url e ter o status final da transação (a requisição pode demorar um pouco).

```
>>> transaction = Transaction(
...     amount='10000',
...     card_id='card_ci3xq3kyu0000yd16rihoplu6',
...     customer=customer,
...     metadata=metadata,
...     soft_descriptor='Pagamento 1'
... )
```

```
>>> transaction.charge()
>>> transaction.data
{u'date_updated': u'2014-12-23T20:13:36.000Z', u'ip': u'179.179.108.26', u'boleto_barcode': None, u'
>>> transaction.data['status']
u'paid'
```

1.2.4 Criando uma transação usando boleto

Para boleto, é interessante sempre usar o postback_url, para ser notificado quando o mesmo for pago.

```
>>> transaction = Transaction(
...     amount='10000',
...     payment_method='boleto',
...     customer=customer,
...     metadata=metadata,
...     postback_url='http://requestb.in/18j98r31',
... )
>>> transaction.charge()
>>> transaction.data
{u'date_updated': u'2014-12-23T20:19:56.000Z', u'ip': u'179.179.108.26', u'boleto_barcode': u'1234 56
```

1.2.5 Consultando os dados de uma transação

Para consultar os dados de uma transação, basta passar o id da mesma.

```
>>> transaction = Transaction()
>>> transaction.find_by_id(173715)
>>> transaction.data
{u'date_updated': u'2014-12-23T20:10:04.000Z', u'ip': u'179.179.108.26', u'boleto_barcode': None, u'
```

1.3 Planos e Assinaturas

Antes de continuar, não esqueça de checar a documentação do pagar.me.

Não esqueça de setar sua api_key.

1.3.1 Criando um novo plano

```
>>> from pagarme.subscription import Plan
>>> plan = Plan(
...     amount='10000',
...     days=30,
...     name='Meu Plano',
...     trial_days=10,
...     charges=10,
...     installments=1
... )
>>> plan.create()
>>> plan.data
{u'name': u'Meu Plano', u'color': None, u'object': u'plan', u'days': 30, u'payment_methods': [u'bolet
```

1.3.2 Consultando os dados de um plano

```
>>> plan = Plan()
>>> plan.find_by_id(10849)
>>> plan.data
{u'name': u'Meu Plano', u'color': None, u'object': u'plan', u'days': 30, u'payment_methods': [u'bolet
```

1.3.3 Criando uma assinatura com cartão de crédito

Os passos para criação de uma nova assinatura são bem parecidos com o da transação, considere o card_hash e customer que foram usados nas transações para os próximos exemplos. Lembrando que para as assinaturas você pode usar o metadata da mesma forma.

Usando o card_hash.

```
>>> subscription = Subscription(
...     plan_id=10849,
...     card_hash=card_hash,
...     customer=customer,
...     postback_url='http://requestb.in/y3jcvey3'
... )
>>> subscription.create()
>>> subscription.data
{u'status': u'trialing', u'customer': {u'name': u'John Appleseed', u'gender': None, u'document_numbe
```

Agora usando o card_id.

```
>>> subscription = Subscription(
...     plan_id=10849,
...     card_id='card_ci3xq3kyu0000yd16rihoplu6',
...     customer=customer,
...     postback_url='http://requestb.in/y3jcvey3'
... )
>>> subscription.create()
>>> subscription.data
{u'status': u'trialing', u'customer': {u'name': u'John Appleseed', u'gender': None, u'document_numbe
```

1.3.4 Criando uma assinatura com boleto

Da mesma forma da transação, é uma boa prática sempre usar o postback_url para assinaturas com boleto.

```
>>> subscription = Subscription(
...     plan_id=10849,
...     payment_method='boleto',
...     customer=customer,
...     postback_url='http://requestb.in/y3jcvey3'
... )
>>> subscription.create()
>>> subscription.data
{u'status': u'trialing', u'customer': {u'name': u'John Appleseed', u'gender': None, u'document_numbe
```

1.3.5 Consultando dados de uma assinatura

```
>>> subscription = Subscription()
>>> subscription.find_by_id(12243)
>>> subscription.data
{u'status': u'trialing', u'customer': {u'name': u'John Appleseed', u'gender': None, u'document_numbe
```

1.3.6 Cancelando uma assinatura

```
>>> subscription = Subscription()
>>> subscription.find_by_id(12243)
>>> subscription.cancel()
>>> subscription.data
{u'status': u'canceled', u'customer': {u'name': u'John Appleseed', u'gender': None, u'document_numbe
>>> subscription.data['status']
u'canceled'
```

1.4 POSTBack

Antes de continuar, não esqueça de checar a documentação do pagar.me.

Não esqueça de setar sua api_key.

1.4.1 Validando a origem de um POSTback

Para validar, basta passar um dicionário com os campos que foram passados via POSTback.

```
>>> from pagarme.postback import PostBack
>>> postback_data = {
...     'old_status': 'waiting_payment',
...     'event': 'transaction_status_changed',
...     'desired_status': 'paid',
...     'fingerprint': 'b4f0bc081343830fddedd61c6996dcb608df6d1c',
...     'object': 'transaction',
...     'current_status': 'paid',
...     'id': 173675
... }
>>> postback = PostBack(postback_data)
>>> postback.is_valid()
True
```

CAPÍTULO 2

Autor

- Allisson Azevedo

CAPÍTULO 3

Principais características

- Python: 2.6, 2.7, 3.2, 3.3 e 3.4.
- Processamento de requisições http/https usando a biblioteca requests.
- Excelente cobertura de testes (> 90%).
- Documentação com exemplos práticos.

CAPÍTULO 4

Links

- Github
- Travis CI
- Coveralls