Cookbook

A short tutorial for using the yope.io payment API.

Creation of the account

in order to use the payment API, you need to create an account.

The creation of an account is very simple. You will be asked your name, an valid email address, a password and a name and a description for your first wallet.

Withdraw your bitcoins from your account's balance is simple and you can specify on which wallet to receive when you request them.

Go here and insert your data. You will receive an email to confirm your email address.

Once that your email address has been confirmed, your account is ready to be used.

Getting your API Key

With your account credential, you can now request an API Key for your application.

The key will allow you to access all the available operation.

To request a key, send a request for a token:

~~POST http://api.yope.io/oauth/token?username={your_email}&password={your_password}&grant_type=password~~

```
POST http://api.yope.io/authenticate?username={your_email}&password={your_password}
```

You need to do this operation only once.

Create your wallets

With the creation of your account, a first wallet is created for you, with the name and the description you provided, or with a default name and description.

You can create as many wallets as you want.

If you have different applications and you want to keep them separated, you can create one wallet per each.

If you have one application, but different items on sale, you can create a wallet for each item.

There is no limit to the ways you can use your wallets.

We differentiate between two kind of wallets:

1. internal
2. external

Internal wallets are used to keep track of all the movement that will happens within your set of wallets. They are not directly associated with a Blockchain wallet, and they can communicate between each other. Internal wallets are the ones where your customer will pay your bitcoins.

External wallets are used for withdraw the bitcoins in your external wallet. They are associated with a hash key, and in a way are a virtual representation in the yope API of your real Bitcoin wallets. you do not need to have one. you can create one every time you want to receive your bitcoins.

Create an internal wallet

You have already a wallet created when you opened your account, but let's say you need a new wallet for a your new application called TIC-TAC-TOE

So now we will create a new wallet called "TIC-TAC-TOE".

To create an internal wallet you need to post the following request:

```
POST http://api.yope.io/wallets
{
    "name": "TIC-TAC-TOE",
    "description": "a wallet for TIC-TAC-TOE game"
}
```

using your API key as Authorization header:

Header	Value
`Content-Type`	`application/json`
`Authorization`	`Bearer your_api_key`

The name of your wallet must be unique within the set of your wallets.

You will use the name to refer to your wallet. For this reason, we advice to use short names, possibly with no spaces into it.

The response from the server is given below:

```
"body": {
    "id": 47,
    "name": "TIC-TAC-TOE",
    "description": "a wallet for TIC-TAC-TOE game"
    "balance": 0,
    "availableBalance": 0,
    "creationDate": 1445279274274
  }
```

So now you have a new wallet. Of course, it will have a `balance` of 0 mBTC as there are yet no movements.

As you noticed, there is a `balance` and there is also an `availableBalance`. As the wallet is new, they are both 0.

The `balance` registers the bitcoins of all the movements that have occurred on the wallet, while the `availableBalance` does register only the movements that have been completed. Later on we will explain better the concept.

Create an external wallet

As written above, you do not really need an external wallet, as you can create one whenever you need to withdraw some bitcoins.

But having an external wallet is a more convenient way to retrieve your bitcoins, as you will not need to pass each time the hash key of your wallet, but just the name of the wallet.

So now we will create a new wallet called "MY_MAIN_WALLET".

To create an internal wallet you need to post the following request:

```
POST http://api.yope.io/wallets
{
    "name": "MY_MAIN_WALLET",
    "description": "a wallet for getting bitcoins",
    "walletHash": "mfqJTEMVvVbbh55Rg1FBmdjz5GUVdJxrof"
 }
```

using your API key as Authorization header:

Header	Value

Content-Type	application/json
Authorization	Bearer your_api_key

Again, the name of your wallet must be unique within the set of your wallets.

The response from the server is given below:

```
{
  "header": {
    "success": true,
    "status": 201
  },
  "body": {
    "id": 48,
    "name": "MY_MAIN_WALLET",
    "description": "a wallet for getting bitcoins",
    "walletHash": "mfqJTEMVvVbbh55Rg1FBmdjz5GUVdJxrof"
    "balance": 0,
    "availableBalance": 0,
    "creationDate": 1445280446123
  }
}
```

Your new wallet is now ready to be used.

DEPOSIT

So, now that you have a wallet, you are ready to accept payment from your customer.

A payment is what we call a DEPOSIT: your customer will deposit a quantity of mBTC in your internal wallet.

In order for your customer to do so, you need to provide him/her with the address of your wallet.

Your wallet has no hash key associated, but it will be given one each time it will need to accept a payment.

To request a key, you need to open a transaction.

So, let's say you want to receive a payment of 10 mBTC for your customer Jo, that want to buy some extra time on your app TIC-TAC-TOE, associated with your wallet TIC-TAC-TOE.

You will open a transaction for Jo, sending the following request:

```
POST http://api.yope.io/transactions
{
  "source":{"name":"Jo"},
  "destination":{"name":"TIC-TAC-TOE"},
  "amount": 10.0,
  "reference": "extra time",
  "description": "10 secs extra time"
}
```

using your API key as Authorization header:

Header	Value
Content-Type	application/json

Authorization	Bearer your_api_key

The request above will instruct the service to create a new transaction between a source wallet named Jo and a destination wallet named TIC-TAC-TOE (your wallet) for an amount of 10 mBTC.

Reference and description are fields you can use for your internal logging.

The response from the server is given below:

```
{
  "header": {
    "success": true,
    "status": 201
  },
  "body": {
    "id": 38,
    "receiverHash": "mrpKCza6ehimpgup4VLKMRB8RZtrPBP7aW",
    "source": {
        "id": 49,
        "name": "Jo"
    },
    "destination": {
        "id": 47,
        "name": "TIC-TAC-TOE",
        "description": "a wallet for TIC-TAC-TOE game"
    },
    "type": "DEPOSIT",
    "status": "PENDING",
    "amount": 10.0,
    "reference": "extra time",
    "description": "10 secs extra time",
    "creationDate": 1445282009771,
    "qr": "http://api.yope.io/images/mrpKCza6ehimpgup4VLKMRB8RZtrPBP7aW.png"
  }
}
```

Let's see what you have received.

A new DEPOSIT transaction has been created, between a new wallet named Jo, and your existing wallet TIC-TAC-TOE.

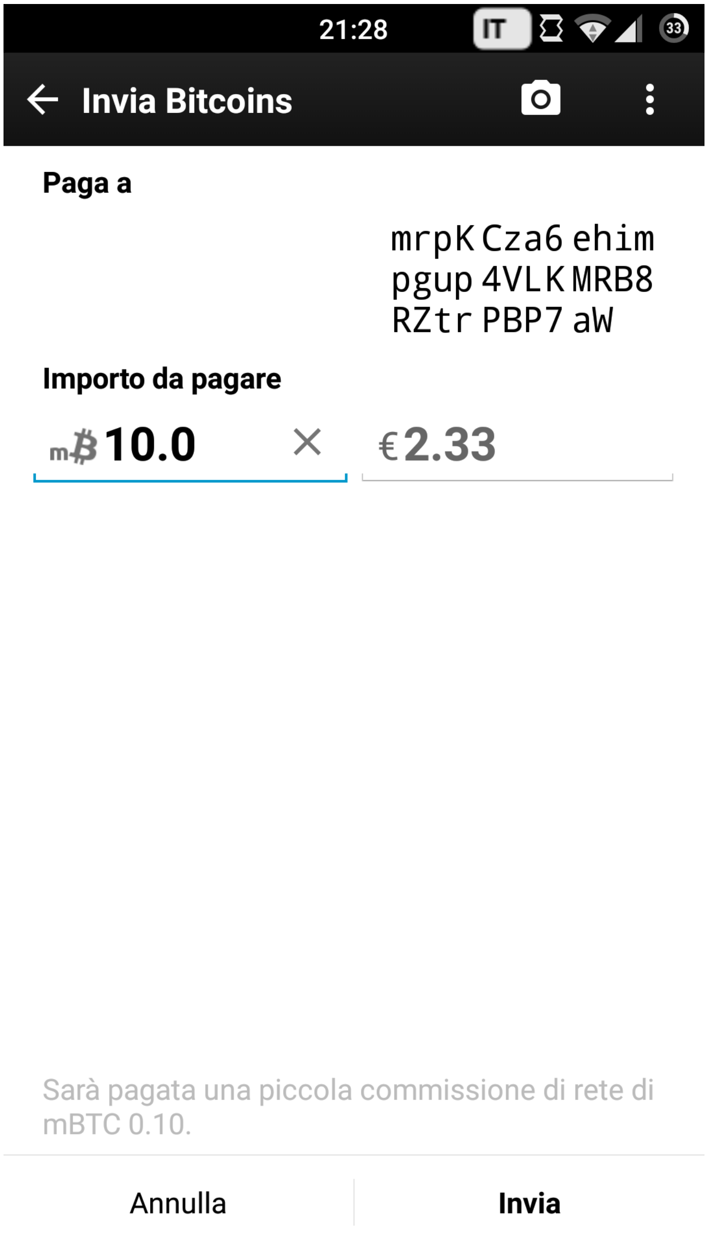
The system detect a DEPOSIT as the transaction is happening between one of your internal wallet (TIC-TAC-TOE) and a wallet that does not exist into the system. The wallet will be created and treated as a TRANSIT wallet.

The most important bit of the transaction is the `receiverHash` and the `qr` url.

These are the value that you have to pass to your customer for completing the payment.

The `receiverHash` is a single-use hash and it is associated to this transaction only for a given time.

The `qr` represent the link to a QR image that contains all the information for this payment:



Paga a

mrpK Cza6 ehim
pgup 4VLK MRB8
RZtr PBP7 aW

Importo da pagare

m฿ 10.0 €2.33

Sarà pagata una piccola commissione di rete di mBTC 0.10.

Your customer can scan the QR code, or copy and paste the receiver hash and proceed to payment.

The transaction is now in `PENDING` status. This means that no movement is still happened. It will stay in this status until your customer perform the payment. Until now the transaction has no effect on your wallet.

You can check the status of the transaction by doing the request:

```
GET POST http://api.yope.io/transactions/38
```

The transaction will look like the following:

```
{
  "header": {
    "success": true,
    "status": 201
  },
  "body": {
    "id": 38,
    "transactionHash":
"292c9745ccf62f53e48d7b98cf5841ee01ac158998a9253008b79d17e5aa04f6",
    "senderHash": "n2H2woqtkHzYjbka6ZAFH3sxk8QR1op6Wb",
 "source": {
        "id": 49,
        "name": "Jo"
    },
    "destination": {
        "id": 47,
        "name": "TIC-TAC-TOE",
        "description": "a wallet for TIC-TAC-TOE game"
    },
    "type": "DEPOSIT",
 "reference": "extra time",
    "description": "10 secs extra time",
    "status": "ACCEPTED",
    "amount": 10.0,
 "balance": 9.9,
    "blockchainFees": 0.1,
    "fees": 0,
    "creationDate": 1445282009771,
    "acceptedDate": 1445282069000,
  }
}
```

Let's see what you have received.

field	value	description
`status`	`ACCEPTED`	your customer has made a payment to the key you have provided to him, and the blockchain has received and accepted the payment, generating a new transaction.
`amount`	`10.0`	the amount of the payment
`balance`	`9.9`	the actual bitcoins received, that is the amount minus the blockchain fees, if any
`blockchainFees`	`0.1`	the fees for the mining of your transaction, paid to blockchain miners.
`fees`	`0`	the Yope fees, if any
`creationDate`	`1445282009771`	when your transaction has been created, Mon Oct 19 2015 21:13:29 GMT+0200 (CEST)

acceptedDate	1445282069000	when your transaction has been accepted, Mon Oct 19 2015 21:14:29 GMT+0200 (CEST)
transactionHash	292c9745ccf62f53e48d7b98cf5841ee01ac158998a9253008b79d17e5aa04f6	the transaction hash on the blockchain. you can use this value to check your transaction directly
senderHash	n2H2woqtkHzYjbka6ZAFH3sxk8QR1op6Wb	the wallet key address of the sender, in this case your customer.

Once you have created a transaction, its state can change. The possible changes are:

- PENDING the transaction has not been yet processed; your customer has not yet made a payment, or the payment has not been yet notified to our system
- ACCEPTED the transaction has been processed after your customer has made a payment and the payment has been notified to our system; in this status, the payment has not yet any confirmation
- EXPIRED the transaction has expired, as the customer has not made any payment, or the payment has not been yet notified to our system after a given interval
- DENIED the transaction has been denied
- FAILED the transaction has failed for internal server error
- COMPLETED the transaction has been successfully completed, the payment has been confirmed in the blockchain and now the full balance is available to your wallet.

As we mentioned before, a wallet has a balance and a availableBalance.

- the balance represent the possible number of coins available in your wallet, as given by the sum of all the ACCEPTED and COMPLETED transactions that occurred on the wallet
- the availableBalance represent the actual number of coins available in your wallet, as given by the sum of all the COMPLETED transactions that occurred on the wallet

As soon as the transaction has been ACCEPTED, the payment went through, and you can proceed with your app transaction and provide the services requested to your customer.

WITHDRAW

So, now a few of your customer have paid services through the yope payment service, and you want to use that payments to buy some stuff on Amazon.

It is time to withdraw your coins from the payment account to your external wallet.

To do so, you need to create a transaction with the following request:

```
POST http://api.yope.io/transactions
{
  "source":{"name":"TIC-TAC-TOE"},
  "destination":{
 "name":"my-wallet"
 "walletHash":"2N5KpWm828A7sdXTPuNyYNtx2Wmo6Jm6r5Z"
  },
  "amount": 10.0,
  "reference": "ref-663",
  "description": "pay time"
}
```

using your API key as Authorization header:

Header	Value
Content-Type	application/json

`Authorization`	`Bearer your_api_key`

```
{
  "header": {
    "success": true,
    "status": 201
  },
  "body": {
    "id": 21,
    "source": {
        "id": 47,
        "name": "TIC-TAC-TOE",
        "description": "a wallet for TIC-TAC-TOE game"
    },
    "destination": {
      "id": 20,
      "walletHash": "2N5KpWm828A7sdXTPuNyYNtx2Wmo6Jm6r5Z",
      "name":"my-wallet"
    },
    "type": "WITHDRAW",
    "reference": "ref-663",
    "status": "ACCEPTED",
    "description": "pay time",
    "amount": 10.0,
    "fees": 0,
    "creationDate": 1445783572423
  }
}
```

using your API key as Authorization header:

The mechanism is similar to the one explained in the section above for the DEPOSIT, but this time you will receive the transaction directly in the `ACCEPTED` state, as our system will send the money as soon as it does receive a valid transaction.