postme Documentation

Release 0.1-beta

Matthieu Martinot

Jun 05, 2017

Contents

CHAPTER 1

Overview

Postme.io

From duplicative invoice to single invoice.

Postme.io combines blockchain technology for billing. We improve data security and transparency of payments in the business relationship. Billing becomes simpler, cheaper, faster and more transparent.

Postme API

Postme API allows your favorite accounting system to communicate with Postchain, to reach a new level in data consistency.

Postchain

Postchain is a Blockchain infrastructure that routes invoices.

CHAPTER 2

Terminology

There is some specialized terminology associated with Postme. To get started, you should at least know what we mean by this terms.

CCO Contract

A CCO (Company Chief Officer) contract link a user and its company. It attests that the user is a company's officer.

CBS Contract

A CBS (Company Billing System) contract link an application (your billing system) to the company. A CCO contract is required to create a CBS contract into Postchain.

CBST Contract

A CBST (Company Billing System Third) contract link an application to third companies. A CCO contract is required to create a CBST contract into Postchain.

Editor Node

(in progress)

Chartered Node

(in progress)

Organization Node

(in progress)

Peer Node

(in progress)

Cluster

(in progress)

Consortium

(in progress)

CHAPTER 3

Getting started

Prerequisites

Create an account on postme.io

First thing you need to do is creating an account on postme.io.

- Go to https://api.postme.io
- Click on “Create an account”
- Activate your email address

Then create or import your public and private keypair to do any action on Postchain Keypairs on postme are generated using the EDCSA 58bits ... algorithm.. see Cryptography.

In your account section you will find a keypair generator to create new keypair.

Entreprise account activation

An enterprise account is necessary to create a new application.

- Go to https://api.postme.io/bo/company
- Fill the form with your company information
- Save

As you are the first user to have created this enterprise account. You are the default contact for this company. Then we create a CCO contract that is timestamped in Postchain. The checkbox indicating that the user is a corporate officer must be validated.

Create a new application

The application will be linked with your company, allowing you to send your company's invoices.

If you want to send your invoices into Postchain over the Postme API with your own invoicing system.

- From the dashboard, go to "My apps"
- Click on "New App"
- Enter a title for your app (i.e. "My invoicing system")
- Save

Once your application is created, you can generate App ID/key pairs to get API access. Then we create a CBS contract that is timestamped in Postchain. The checkbox indicating that the application is represented by the company.

- Click on "Generate New Key"
- Enter a name for the new generated key (i.e. "Test")
- If you want to use these credentials for test purpose, check "Sandbox Mode"

Use your account as an editor

Necessary to send invoices from your customers.

(In progress)

Using Postme API

Authentication

POST /api/login

Example request:

```
POST /api/login HTTP/1.1
Host: example.com
Accept: application/json
```

Example response:

```
HTTP/1.1 200 OK
Vary: Accept
Content-Type: application/json
```

JSON Parameters

- **app_id** (*string*) – your APP_ID
- **app_secret** (*string*) – your APP_SECRET

Request Headers

- Accept – `application/json`

Response Headers

- Content-Type – `application/json`

Status Codes

- 200 OK – Authentication succeeded
- 401 Unauthorized – Wrong credentials

Root URL

If you send an HTTP GET request to the Postme API Root URL e.g. `https://api.postme.io/api`, then you should get an HTTP response with something like the following in the body:

(In Progress)

```
HTTP/1.1 200 OK
Content-Type: application/json

{
      "_links" : {
        "api_v1": "https://api.postme.io/api/v1/",
        "docs": "https://docs.postme.io/",
        "..."
      },
      "third_application": {
            "app_id": "5349162981239685"
      },
      "version": "0.1.0"
}
```

Root Endpoint

If you send an HTTP GET request to the Postme API Root Endpoint e.g. `https://api.postme.io/api/v1`, then you should get an HTTP response that allows you to discover the Postme API endpoints :

(In Progress)

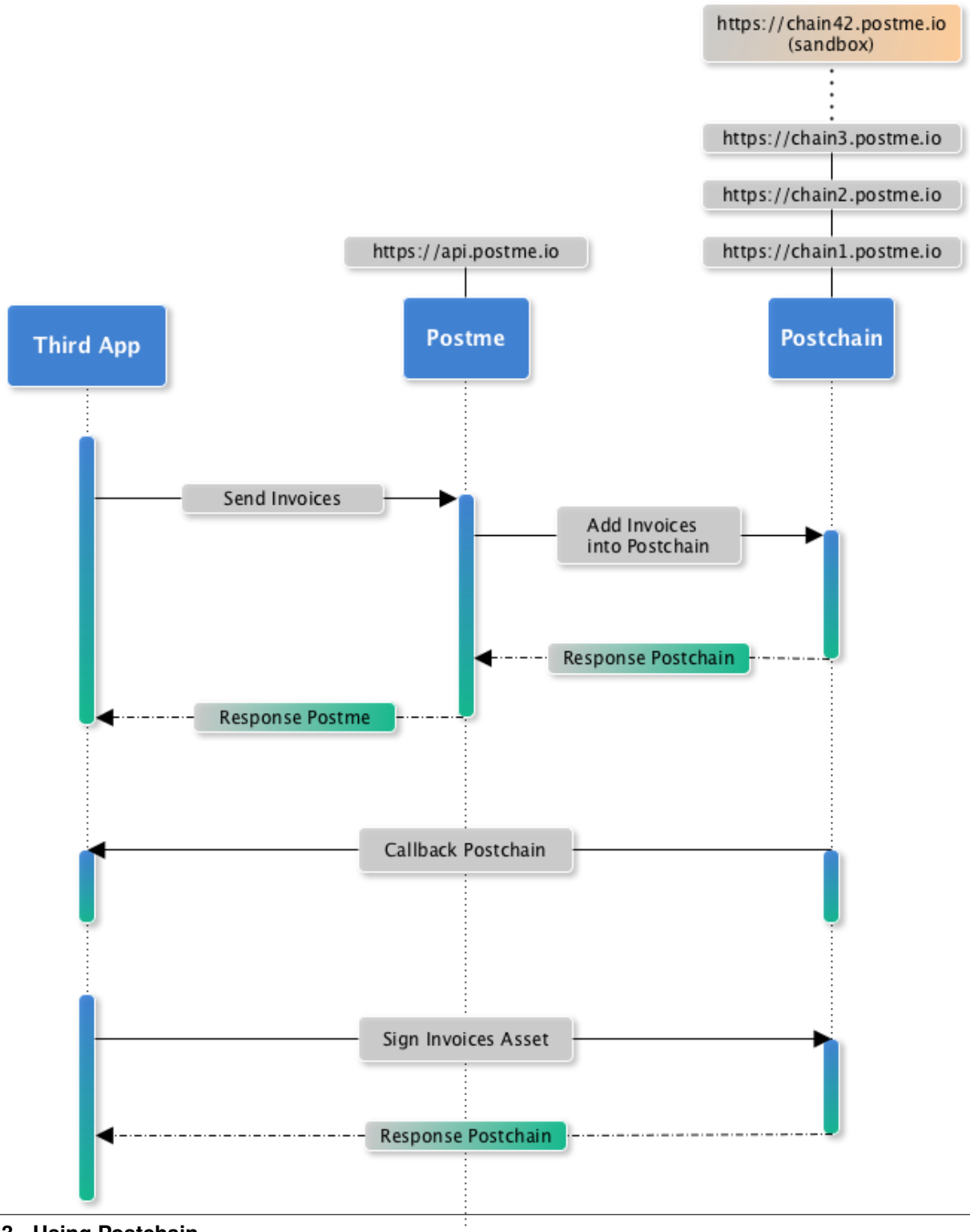
```
HTTP/1.1 200 OK
Content-Type: application/json

{
      "_links" : {
        "self": "https://api.postme.io/api/v1/",
        "invoices": "https://api.postme.io/api/v1/invoices/",
        "..."
      }
}
```

Using Postchain

Create a new private/public keys

Behaviour



CHAPTER 4

Postme HTTP API

Companies

POST /api/company

Push a new company.

Example request:

```
POST /api/company HTTP/1.1
Host: example.com
Content-Type: application/json

{
   "name": "My Organization",
   "address":  {
      "address_1": "456 Street Name",
      "address_2": "",
      "postal_code": "69003",
      "city": "Lyon",
      "country_code": "FR"
   },
   "identification": {
      "siren_id": "654654654",
      "vat_id": "FR27654654654",
      "siret_id": "65465465400045"
   }
}
```

Example response:

```
HTTP/1.1 201 Created
Vary: Accept
Content-Type: application/json
```

Request Headers

- Accept – `application/json`
- Authorization – `Bearer [TOKEN]`

Response Headers

- Content-Type – `application/json`

Status Codes

- 201 Created – Company created
- 400 Bad Request – Invalid params
- 401 Unauthorized – Invalid Authorization Token

Invoices

POST /api/invoices

Push a new invoice.

Example request:

```
POST /api/invoices HTTP/1.1
Host: example.com
Content-Type: application/json
```

Example response:

```
HTTP/1.1 202 Accepted
Vary: Accept
Content-Type: application/json
```

JSON Parameters

- **invoice** (*object-invoice*) – the invoice

Request Headers

- Accept – `application/json`
- Authorization – `Bearer [TOKEN]`

Response Headers

- Content-Type – `application/json`

Status Codes

- 202 Accepted – Invoice successfully sent to postme
- 400 Bad Request – Invalid params
- 401 Unauthorized – Invalid Authorization Token

Parties

POST /api/parties

Push a new party.

Example request:

```
POST /api/parties HTTP/1.1
Host: example.com
Content-Type: application/json

{
   "app_id": "456456",
   "app_reference": "CUSTOMER-145",
   "type": "individual",
   "person": {
      "name": "John Doe",
      "email": "john.doe@gmail.com",
      "phone": "+33601010101",
      "address": { }
   }
}
```

Example response:

```
HTTP/1.1 201 Created
Vary: Accept
Content-Type: application/json
```

Request Headers

- Accept – `application/json`
- Authorization – `Bearer [TOKEN]`

Response Headers

- Content-Type – `application/json`

Status Codes

- 201 Created – Party created
- 400 Bad Request – Invalid params
- 401 Unauthorized – Invalid Authorization Token
- 409 Conflict – Another party with ID `app_id` already exists

GET /api/parties

Get all parties

Example request:

```
GET /api/parties HTTP/1.1
Host: example.com
```

Example response:

```
HTTP/1.1 200 OK
Content-Type: application/json

[
   {
      "id": "00b1c348-260b-11e7-80e3-c6514258f3b9",
      "app_id": "456456",
```

```
        "app_reference": "CUSTOMER-145",
        "type": "individual",
        "person": {
            "name": "John Doe",
            "email": "john.doe@gmail.com",
            "phone": "+33601010101",
            "address": { }
        }
    },
    {
        "id": "ffb057b4-260a-11e7-80e3-c6514258f3b9",
        "app_id": "456456",
        "app_reference": "CUSTOMER-145",
        "type": "individual",
        "person": {
            "name": "John Doe",
            "email": "john.doe@gmail.com",
            "phone": "+33601010101",
            "address": { }
        }
    }
]
```

Request Headers

- Authorization – `Bearer [TOKEN]`

Response Headers

- Content-Type – `application/json`

Status Codes

- 200 OK – A list of parties
- 400 Bad Request – Invalid params
- 401 Unauthorized – Invalid Authorization Token

GET /api/parties/{app_id}

Get the party with the ID `app_id`.

Example request:

```
GET /api/parties/456456 HTTP/1.1
Host: example.com
```

Example response:

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "id": "fde7cc83-260a-11e7-80e3-c6514258f3b9",
    "app_id": "456456",
    "app_reference": "CUSTOMER-145",
    "type": "individual",
    "person": {
        "name": "John Doe",
        "email": "john.doe@gmail.com",
```

```
        "phone": "+33601010101",
        "address": { }
    }
}
```

Request Headers

- Authorization – `Bearer [TOKEN]`

Response Headers

- Content-Type – `application/json`

Status Codes

- 200 OK – A Party with that ID was found
- 400 Bad Request – Invalid params
- 401 Unauthorized – Invalid Authorization Token

CHAPTER 5

Data Models

Address

Structure

```
{
        "address_1": "",
        "address_2": "",
        "postal_code": "69003",
        "city": "Lyon",
        "country_code": "FR"
}
```

Attributes

- `address_1` : Required. *Address 1*. type: string. format: alphanumeric, max length: 255 char.
- `address_2` : Optional. *Address 2*. type: string. format: alphanumeric, max length: 255 char.
- `postal_code` : Required. *Postal Code*. type: string. format: alphanumeric, max length: 16 char.
- `city` : Required. *Postal Code*. type: string. format: alphanumeric, max length: 64 char.
- `country_code` : Required. *Country code of the address*. type: string. format: 2 digits As defined by the ISO 3166-1 alpha-2.

Company

Structure

```
{
        "name": "My Organization",
        "address": {},
        "identification": {
```

```
                "siren_id": "654654654",
                "vat_id":   "FR27654654654",
                "siret_id": "65465465400045",
        }
}
```

Attributes

- `name` Required. *The name of the company*. type: string. format: alphanumeric.
- `address` Required. *Address of the company*. type: object *Address*.
- **`identification` Required. *Identification of the company*. type: object. *Required fields are based on company's address c***
  - `siren_id` 'FR': Required. *SIREN Identification Number of the company (for french companies only)*. type: string. format: numeric.
  - `vat_id` Optional. *VAT Identification Number of the company*. type: string. format: alphanumeric.
  - `siret_id` Optional. *SIRET Identification Number of the company branch (for french companies only)*. type: string. format: numeric.

Contact

Structure

```
{
        "name": "John Doe (accounting service)",
        "email": "john.doe@company.com",
        "phone": "+33601010101",
        "address": { }
}
```

Attributes

- `name` Required. *Name of the contact*. type: string. format: alpha.
- `email` Required. *Email of the contact*. type: string. format: valid email.
- `phone` Optional. *Phone number of the contact*. type: string. format: valid phone number as defined by E.164, the international public telecommunication numbering plan.
- `address` Optional. *Address of the contact*. type: object *Address*.

Invoice

Structure

```
{
        "app_invoice_id": "1234",
        "seller_party": {
                "app_party_id": "123123",
                "type": "professional",
                "company": { },
                "contacts": [ ]
        },
```

```
        "buyer_party": {
                "app_party_id": "456456",
                "type": "professional",
                "company": { },
                "contacts": [ ],
                "person": { }
        },
        "delivery_address": { },
        "reference": "INV201701010004",
        "description": "My first invoice",
        "issue_date": "2017-01-01",
        "invoice_type_code": "S",
        "currency_code": "EUR",
        "taxes": [
                {
                        "tax_rate": 5.5,
                        "total": 200,
                        "total_taxes": 11,
                        "total_due": 211
                },
                {
                        "tax_rate": 19.6,
                        "total": 1000,
                        "total_taxes": 196,
                        "total_due": 1196
                }
        ],
        "total": 1200,
        "total_taxes": 207,
        "total_due": 1407,
        "terms": {
                "due_date": "2017-02-01",
                "payment": "Before Jun 31st",
                "vat": "Not applicable"
        },
        "lines": [],
        "journal_entries": [],
        "notes": "Some free text..."
}
```

Attributes

- `app_invoice_id` Required. *Invoice identifier of the third party application.* type: string. format: alphanumeric.
- `seller_party` Required. *The seller party of the invoice.*
- `seller_party[app_party_id]` Required. *Party identifier of the third party application.* type: string. format: alphanumeric.
- `seller_party[type]` Required. type: string. values: *'professional'*.
- `seller_party[company]` Required. *The company of the seller party.* type: object *Company*.
- `seller_party[contacts]` Required. *Administrative contacts of the seller party.* type: Array<object *Contact*>.
- `buyer_party` Required. *The buyer party of the invoice.*
- `buyer_party[app_party_id]` Required. *Party identifier of the third party application.* type: string. format: alphanumeric.

- `buyer_party[type]` Required. type: string. values: *'professional'* | *'institutional'* | *'individual'*.
- `buyer_party[company]` Required if type is *professional* (none otherwise). *The company of the buyer party*. type: object *Company*.
- `buyer_party[contacts]` Required if type is *professional* (none otherwise). *Administrative contacts of the buyer party*. type: Array<object *Contact*>.
- `buyer_party[person]` Required if type is *individual* (none otherwise). type: object *Person*.
- `delivery_address` Optional. *Invoice's Delivery Address.* type: object *Address*.
- `reference` Required. *Invoice reference number.* type: string. format: alphanumeric.
- `description` Optional. *Invoice description.* type: string. format: alphanumeric.
- `issue_date` Required. type: string. format: date.
- `invoice_type_code` Required. *Type of the invoice*. type: char. value:s *'S'* (standard) | *'C'* (credit note).
- `currency_code` Required. *Currency used in invoice format*. type: string. format: 3 digits as defined by [ISO 4217](https://www.iso.org/iso-4217-currency-codes.html).
- `taxes` Optional. *An array of taxes with a different rate for the invoice*.
- `total` Required. *Total amount of the invoice before taxes*. type: decimal.
- `total_taxes` Required. *Taxes amount of the invoice*. type: decimal.
- `total_due` Required. *Total amount of the invoice including taxes*. type: decimal.
- `terms`: Optional. *List of terms. See JSON example for available fields*. type: Array<Dict>.
- `lines` Required. *Invoice lines*. type: Array<object *InvoiceLine*>.
- `journal_entries` Optional. *Invoice's journal entries*. type: Array<object *JournalEntry*>.
- `notes`: Optional. *Free text.* type: string.

InvoiceLine

Structure

```
{
        "description": "Food",
        "taxes": [
                {
                        "tax_rate": 5.5,
                        "total": 20,
                        "total_taxes": 1.1,
                        "total_due": 21.1
                },
                {
                        "tax_rate": 19.6,
                        "total": 10,
                        "total_taxes": 1.96,
                        "total_due": 11.96
                }
        ],
        "total": 30,
        "total_taxes": 2.07,
        "total_due": 32.07,
```

```
        "items": []
}
```

Attributes

- `description` Optional. *Free form text*. type: string. format: alphanumeric.
- `taxes` Optional. *An array of taxes with a different rate for the invoice*.
- `total` Required. *Total amount of the invoice line before taxes*. type: decimal.
- `taxes` Required. *Taxes amount of the invoice line*. type: decimal.
- `total_due` Required. *Total amount of the invoice line including taxes*. type: decimal.
- `items` Required. *Line items*. type: Array<object *InvoiceLineItem*>

InvoiceLineItem

Structure

```
{
        "lot_id": "ABCDE12345",
        "description": "Beef steak",
        "quantity": 12.5,
        "unit": "kg",
        "unit_price": 4,
        "total": 50,
        "tax_rate": 5.5,
        "total_taxes": 2.75,
        "total_due": 52.75,
        "journal_entries": []
}
```

Attributes

- `lot_id` Optional. *Item's lot identification number*. type: string. format: alphanumeric
- `description` Required. type: string. format: alphanumeric.
- `quantity` Optional. type: decimal.
- `unit` Optional. type: string. format: alphanumeric
- `unit_price` Optional. type: decimal.
- `total` Required. *Total amount of the invoice line item before taxes*. type: decimal.
- `total_taxes` Required. *Taxes amount of the invoice line item*. type: decimal.
- `total_due` Required. *Total amount of the invoice line item including taxes*. type: decimal.
- `journal_entries` Optional. *Item's journal entries*. type: Array<object *JournalEntry*>.

JournalEntry

Structure

```
{
        "app_journal_id": "2",
        "journal_code": "SA",
        "journal_description": "Sales",
        "account_number": "445710",
        "description": "Collected VAT",
        "debit": 0,
        "credit": 310.54
}
```

Attributes

- `app_journal_id` Optional. *Journal ID of the accounting journal*. type: string. format: alphanumeric.
- `journal_code` Optional. *Journal code of the accounting journal*. type: string. format: alphanumeric.
- `journal_description` Optional. *Journal description of the accounting journal*. type: string. format: alphanumeric.
- `account_number` Required. *Account number for the accounting entry*. type: string. format: alphanumeric.
- `account_description` Optional. *Account description*. type: string. format: alphanumeric
- `debit` Required. *Debit amount*. type: decimal
- `credit` Required. *Credit amount*. type: decimal

Person

Structure

```
{
        "name": "John Doe",
        "email": "john.doe@gmail.com",
        "phone": "+33601010101",
        "address": { }
}
```

Attributes

- `name` Required. *Name of the person*. type: string. format: alpha.
- `email` Required. *Email of the person*. type: string. format: valid email.
- `phone` Optional. *Phone number of the person*. type: string. format: valid phone number as defined by E.164, the international public telecommunication numbering plan.
- `address` Optional. *Address of the person*. type: object *Address*.

CHAPTER 6

KYC Editor

CHAPTER 7

KYC Organization

HTTP Routing Table

/api