



July 4, 2022

Crypto Pay API

[Crypto Pay](#) is a payment system based on [Crypto Bot](#) that allows you to accept payments in crypto and transfer coins to users using our API.

Subscribe to our [News Channel](#) to be the first to know about the latest updates and join the discussion in [our chat](#).

Recent changes

Crypto Pay API 1.5 (11 September, 2024)

- Added the *spend_id* field to the method [getTransfers](#) and class [Transfer](#).

Crypto Pay API 1.4 (21 June, 2024)

- Added the fields *mini_app_invoice_url*, *web_app_invoice_url* to the class [Invoice](#).

Crypto Pay API 1.3 (15 March, 2024)

- Added the ability to get app statistics using the [getStats](#) method and the class [AppStats](#).
- Added the ability to create checks with attachment to the user through the id or his username.

Crypto Pay API 1.2 (24 November, 2023)

- Added the methods [createCheck](#), [deleteCheck](#), [getChecks](#) and the class [Check](#) to manage checks by using the app.
- Added the parameters *currency_type*, *fiat* and *accepted_assets* to the method [createInvoice](#) to create invoices for the amount specified in a fiat currency.
- Added the ability to delete invoices using the method [deleteInvoice](#).
- Added the ability to get the list of transfers using the method [getTransfers](#).

- Added the fields *currency_type*, *fiat*, *paid_asset*, *paid_amount*, *paid_usd_rate*, *paid_fiat_rate*, *fee_asset*, *fee_amount* and *accepted_assets* to the class [Invoice](#).
- The field *fee* in the Webhook update payload is now deprecated, use the new field *fee_amount* in the class [Invoice](#) instead.
- The field *usd_rate* in the Webhook update payload is now deprecated, use the new field *paid_usd_rate* in the class [Invoice](#) instead.
- The field *pay_url* is now deprecated, use the new field *bot_invoice_url* in the class [Invoice](#) instead.
- Added the field *onhold* to the class [Balance](#).

Crypto Pay API 1.1.5 (September 30, 2023)

- Discontinued support for BUSD.

Crypto Pay API 1.1.4 (September 22, 2023)

- Added support for LTC to the mainnet.

Crypto Pay API 1.1.3 (October 3, 2022)

- Added the field *fee* to the class [Invoice](#) to show the amount of charged service fees. Returned only in the Webhook update payload.
- Added the field *usd_rate* to the class [Invoice](#) to show the price of the *asset* in USD. Returned only in the Webhook update payload.

Crypto Pay API 1.1.2 (September 5, 2022)

- Method [transfer](#) is disabled by default for new apps.
- You can set an allowlist of IP addresses under the “Security” button in the app settings.

Crypto Pay API 1.1.1 (July 26, 2022)

- Added support for ETH to the mainnet.

Crypto Pay API 1.1 (February 2, 2022)

- Apps can now send coins to users using the new method [transfer](#).
- Added support for [hidden message](#) in invoices.
- New parameter *expires_in* to set a payment time limit for new invoices.
- Added new *expiration_date* field in [Invoice](#) object.

Note: In order to use the new method *transfer*, you need to [create a new app](#).

Authorizing your app

First, you need to create a new app and get API token. Open [@CryptoBot](#) ([@CryptoTestnetBot](#) for testnet), go to [Crypto Pay](#) and tap **Create App** to get API Token.

All queries to Crypto Pay API must be served over **HTTPS**. Use either [URL query string](#) or *application/json* or *application/x-www-form-urlencoded* or *multipart/form-data* for passing parameters. API Token must be passed in the header parameter `Crypto-Pay-API-Token`. URL must be presented in this form:

`https://pay.crypt.bot/api/%method%`. Example request:

```
GET /api/getMe HTTP/1.1
Host: pay.crypt.bot
Crypto-Pay-API-Token: 123456789:AAzQcZWQqQAbsfgPn0Lr4FHC8Doa4L7KryC
```

Testnet:

Bot: [@CryptoTestnetBot](#)

URL: <https://testnet-pay.crypt.bot/>

Mainnet:

Bot: [@CryptoBot](#)

URL: <https://pay.crypt.bot/>

Getting updates

There are two ways of receiving updates for your app — you can use [getInvoices](#) method to get a list of created invoices or [Webhooks](#) to receive updates in realtime.

Methods

- [getMe](#)
- [createInvoice](#)
- [deleteInvoice](#)
- [createCheck](#)
- [deleteCheck](#)
- [transfer](#)
- [getInvoices](#)
- [getChecks](#)

- [getTransfers](#)
- [getBalance](#)
- [getExchangeRates](#)
- [getCurrencies](#)
- [getStats](#)

Types

- [Invoice](#)
- [Transfer](#)
- [Check](#)
- [Balance](#)
- [ExchangeRate](#)
- [AppStats](#)

Available methods

We support **GET** and **POST** HTTP methods. Response contains a JSON-object which always has the Boolean field `ok`. If `ok` equals *true*, the request was successful, and the result of the query can be found in `result` field. In case of an unsuccessful request, `ok` equals *false*, and the error is explained in `error` field (e.g. `PARAM_SHORT_NAME_REQUIRED`). All queries must be made using UTF-8.

| **Tip:** Use [Big.js](#) or another lib to work with big numbers.

getMe

Use this method to test your app's authentication token. Requires no parameters. On success, returns basic information about an app.

createInvoice

Use this method to create a new invoice. On success, returns an object of the created [invoice](#).

- **currency_type** (String)
Optional. Type of the price, can be "crypto" or "fiat". Defaults to *crypto*.
- **asset** (String)
Optional. Required if *currency_type* is "crypto". Cryptocurrency alphabetic code. Supported assets: "USDT", "TON", "BTC", "ETH", "LTC", "BNB", "TRX" and "USDC".

- **fiat** (String)
Optional. Required if *currency_type* is “fiat”. Fiat currency code. Supported fiat currencies: “USD”, “EUR”, “RUB”, “BYN”, “UAH”, “GBP”, “CNY”, “KZT”, “UZS”, “GEL”, “TRY”, “AMD”, “THB”, “INR”, “BRL”, “IDR”, “AZN”, “AED”, “PLN” and “ILS”.
- **accepted_assets** (String)
Optional. List of cryptocurrency alphabetic codes separated comma. Assets which can be used to pay the invoice. Available only if *currency_type* is “fiat”. Supported assets: “USDT”, “TON”, “BTC”, “ETH”, “LTC”, “BNB”, “TRX” and “USDC” (and “JET” for testnet). Defaults to all currencies.
- **amount** (String)
Amount of the invoice in float. For example: `125.50`
- **description** (String)
Optional. Description for the invoice. User will see this description when they pay the invoice. Up to 1024 characters.
- **hidden_message** (String)
Optional. Text of the message which will be presented to a user after the invoice is paid. Up to 2048 characters.
- **paid_btn_name** (String)
Optional. Label of the button which will be presented to a user after the invoice is paid. Supported names:
`viewItem` – “View Item”
`openChannel` – “View Channel”
`openBot` – “Open Bot”
`callback` – “Return”
- **paid_btn_url** (String)
Optional. Required if *paid_btn_name* is specified. URL opened using the button which will be presented to a user after the invoice is paid. You can set any callback link (for example, a success link or link to homepage). Starts with *https* or *http*.
- **payload** (String)
Optional. Any data you want to attach to the invoice (for example, user ID, payment ID, ect). Up to 4kb.
- **allow_comments** (Boolean)
Optional. Allow a user to add a comment to the payment. Defaults to *true*.
- **allow_anonymous** (Boolean)
Optional. Allow a user to pay the invoice anonymously. Defaults to *true*.

- **expires_in** (Number)

Optional. You can set a payment time limit for the invoice in seconds. Values between 1-2678400 are accepted.

deleteInvoice

Use this method to delete invoices created by your app. Returns *True* on success.

- **invoice_id** (Number)

Invoice ID to be deleted.

createCheck

Use this method to create a new check. On success, returns an object of the created [check](#).

- **asset** (String)

Cryptocurrency alphabetic code. Supported assets: "USDT", "TON", "BTC", "ETH", "LTC", "BNB", "TRX" and "USDC" (and "JET" for testnet).

- **amount** (String)

Amount of the check in float. For example: `125.50`

- **pin_to_user_id** (Number)

Optional. ID of the user who will be able to activate the check.

- **pin_to_username** (String)

Optional. A user with the specified username will be able to activate the check.

deleteCheck

Use this method to delete checks created by your app. Returns *True* on success.

- **check_id** (Number)

Check ID to be deleted.

transfer

Use this method to send coins from your app's balance to a user. On success, returns completed [transfer](#). This method must first be enabled in the security settings of your app. Open [@CryptoBot](#) ([@CryptoTestnetBot](#) for testnet), go to [Crypto Pay](#) → *My Apps*, choose an app, then go to *Security* → *Transfers...* and tap **Enable**.

- **user_id** (Number)

User ID in Telegram. User must have previously used [@CryptoBot](#) ([@CryptoTestnetBot](#) for testnet).

- **asset** (String)

Cryptocurrency alphabetic code. Supported assets: "USDT", "TON", "BTC", "ETH", "LTC", "BNB", "TRX" and "USDC" (and "JET" for testnet).

- **amount** (String)
Amount of the transfer in float. The minimum and maximum amount limits for each of the supported assets roughly correspond to 1-25000 USD. Use [getExchangeRates](#) to convert amounts. For example: 125.50
- **spend_id** (String)
Random UTF-8 string unique per transfer for idempotent requests. The same `spend_id` can be accepted only once from your app. Up to 64 symbols.
- **comment** (String)
Optional. Comment for the transfer. Users will see this comment in the notification about the transfer. Up to 1024 symbols.
- **disable_send_notification** (Boolean)
Optional. Pass *true* to not send to the user the notification about the transfer. Defaults to *false*.

getInvoices

Use this method to get invoices created by your app. On success, returns array of [Invoice](#).

- **asset** (String)
Optional. Cryptocurrency alphabetic code. Supported assets: "USDT", "TON", "BTC", "ETH", "LTC", "BNB", "TRX" and "USDC" (and "JET" for testnet). Defaults to all currencies.
- **fiat** (String)
Optional. Fiat currency code. Supported fiat currencies: "USD", "EUR", "RUB", "BYN", "UAH", "GBP", "CNY", "KZT", "UZS", "GEL", "TRY", "AMD", "THB", "INR", "BRL", "IDR", "AZN", "AED", "PLN" and "ILS". Defaults to all currencies.
- **invoice_ids** (String)
Optional. List of invoice IDs separated by comma.
- **status** (String)
Optional. Status of invoices to be returned. Available statuses: "active" and "paid". Defaults to all statuses.
- **offset** (Number)
Optional. Offset needed to return a specific subset of invoices. Defaults to 0.
- **count** (Number)
Optional. Number of invoices to be returned. Values between 1-1000 are accepted. Defaults to 100.

getTransfers

Use this method to get transfers created by your app. On success, returns array of [Transfer](#).

- **asset** (String)
Optional. Cryptocurrency alphabetic code. Supported assets: "USDT", "TON", "BTC", "ETH", "LTC", "BNB", "TRX" and "USDC" (and "JET" for testnet). Defaults to all currencies.
- **transfer_ids** (String)
Optional. List of transfer IDs separated by comma.
- **spend_id** (String)
Optional. Unique UTF-8 transfer string.
- **offset** (Number)
Optional. Offset needed to return a specific subset of transfers. Defaults to 0.
- **count** (Number)
Optional. Number of transfers to be returned. Values between 1-1000 are accepted. Defaults to 100.

getChecks

Use this method to get checks created by your app. On success, returns array of [Check](#).

- **asset** (String)
Optional. Cryptocurrency alphabetic code. Supported assets: "USDT", "TON", "BTC", "ETH", "LTC", "BNB", "TRX" and "USDC" (and "JET" for testnet). Defaults to all currencies.
- **check_ids** (String)
Optional. List of check IDs separated by comma.
- **status** (String)
Optional. Status of check to be returned. Available statuses: "active" and "activated". Defaults to all statuses.
- **offset** (Number)
Optional. Offset needed to return a specific subset of check. Defaults to 0.
- **count** (Number)
Optional. Number of check to be returned. Values between 1-1000 are accepted. Defaults to 100.

getBalance

Use this method to get balances of your app. Requires no parameters. Returns array of [Balance](#).

getExchangeRates

Use this method to get exchange rates of supported currencies. Requires no parameters. Returns array of [ExchangeRate](#).

getCurrencies

Use this method to get a list of supported currencies. Requires no parameters. Returns a list of fiat and cryptocurrency alphabetic codes.

getStats

Use this method to get app statistics. On success, returns [AppStats](#).

- **start_at** (String)
Optional. Date from which start calculating statistics in ISO 8601 format. Defaults is current date minus 24 hours.
- **end_at** (String)
Optional. The date on which to finish calculating statistics in ISO 8601 format. Defaults is current date.

Available types

Invoice

- **invoice_id** (Number)
Unique ID for this invoice.
- **hash** (String)
Hash of the invoice.
- **currency_type** (String)
Type of the price, can be "crypto" or "fiat".
- **asset** (String)
Optional. Cryptocurrency code. Available only if the value of the field *currency_type* is "crypto". Currently, can be "USDT", "TON", "BTC", "ETH", "LTC", "BNB", "TRX" and "USDC" (and "JET" for testnet).
- **fiat** (String)
Optional. Fiat currency code. Available only if the value of the field *currency_type* is "fiat". Currently one of "USD", "EUR", "RUB", "BYN", "UAH", "GBP", "CNY", "KZT", "UZS", "GEL", "TRY", "AMD", "THB", "INR", "BRL", "IDR", "AZN", "AED", "PLN" and "ILS".
- **amount** (String)
Amount of the invoice for which the invoice was created.

- **paid_asset** (String)
Optional. Cryptocurrency alphabetic code for which the invoice was paid. Available only if *currency_type* is “fiat” and *status* is “paid”.
- **paid_amount** (String)
Optional. Amount of the invoice for which the invoice was paid. Available only if *currency_type* is “fiat” and *status* is “paid”.
- **paid_fiat_rate** (String)
Optional. The rate of the *paid_asset* valued in the *fiat* currency. Available only if the value of the field *currency_type* is “fiat” and the value of the field *status* is “paid”.
- **accepted_assets** (String)
Optional. List of assets which can be used to pay the invoice. Available only if *currency_type* is “fiat”. Currently, can be “USDT”, “TON”, “BTC”, “ETH”, “LTC”, “BNB”, “TRX” and “USDC” (and “JET” for testnet).
- **fee_asset** (String)
Optional. Asset of service fees charged when the invoice was paid. Available only if *status* is “paid”.
- **fee_amount** (Number)
Optional. Amount of service fees charged when the invoice was paid. Available only if *status* is “paid”.
- **fee** (String) (*deprecated*)
Optional. Amount of charged service fees. Available only in the payload of the webhook update (described here for reference).
- **pay_url** (String) (*deprecated*)
Deprecated. URL should be provided to the user to pay the invoice (described here for reference).
- **bot_invoice_url** (String)
URL should be provided to the user to pay the invoice.
- **mini_app_invoice_url** (String)
Use this URL to pay an invoice to the Telegram Mini App version.
- **web_app_invoice_url** (String)
Use this URL to pay an invoice to the Web version of Crypto Bot.
- **description** (String)
Optional. Description for this invoice.
- **status** (String)
Status of the transfer, can be “active”, “paid” or “expired”.

- **created_at** (String)
Date the invoice was created in ISO 8601 format.
- **paid_usd_rate** (String)
Optional. Price of the *asset* in USD. Available only if *status* is “paid”.
- **usd_rate** (String) (*deprecated*)
Optional. Price of the *asset* in USD. Available only in the Webhook update payload.
- **allow_comments** (Boolean)
True, if the user can add comment to the payment.
- **allow_anonymous** (Boolean)
True, if the user can pay the invoice anonymously.
- **expiration_date** (String)
Optional. Date the invoice expires in ISO 8601 format.
- **paid_at** (String)
Optional. Date the invoice was paid in ISO 8601 format.
- **paid_anonymously** (Boolean)
True, if the invoice was paid anonymously.
- **comment** (String)
Optional. Comment to the payment from the user.
- **hidden_message** (String)
Optional. Text of the hidden message for this invoice.
- **payload** (String)
Optional. Previously provided data for this invoice.
- **paid_btn_name** (String)
Optional. Label of the button, can be “viewItem”, “openChannel”, “openBot” or “callback”.
- **paid_btn_url** (String)
Optional. URL opened using the button.

Transfer

- **transfer_id** (Number)
Unique ID for this transfer.
- **spend_id** (String)
Unique UTF-8 string.
- **user_id** (String)
Telegram user ID the transfer was sent to.

- **asset** (String)
Cryptocurrency alphabetic code. Currently, can be “USDT”, “TON”, “BTC”, “ETH”, “LTC”, “BNB”, “TRX” and “USDC” (and “JET” for testnet).
- **amount** (String)
Amount of the transfer in float.
- **status** (String)
Status of the transfer, can only be “completed”.
- **completed_at** (String)
Date the transfer was completed in ISO 8601 format.
- **comment** (String)
Optional. Comment for this transfer.

Check

- **check_id** (Number)
Unique ID for this check.
- **hash** (String)
Hash of the check.
- **asset** (String)
Cryptocurrency alphabetic code. Currently, can be “USDT”, “TON”, “BTC”, “ETH”, “LTC”, “BNB”, “TRX” and “USDC” (and “JET” for testnet).
- **amount** (String)
Amount of the check in float.
- **bot_check_url** (String)
URL should be provided to the user to activate the check.
- **status** (String)
Status of the check, can be “active” or “activated”.
- **created_at** (String)
Date the check was created in ISO 8601 format.
- **activated_at** (String)
Date the check was activated in ISO 8601 format.

Balance

- **currency_code** (String)
Cryptocurrency alphabetic code. Currently, can be “USDT”, “TON”, “BTC”, “ETH”, “LTC”, “BNB”, “TRX” and “USDC” (and “JET” for testnet).

- **available** (String)
Total available amount in float.
- **onhold** (String)
Unavailable amount currently is on hold in float.

ExchangeRate

- **is_valid** (Boolean)
True, if the received rate is up-to-date.
- **is_crypto** (Boolean)
True, if the source is the cryptocurrency.
- **is_fiat** (Boolean)
True, if the source is the fiat currency.
- **source** (String)
Cryptocurrency alphabetic code. Currently, can be "USDT", "TON", "BTC", "ETH", "LTC", "BNB", "TRX" and "USDC".
- **target** (String)
Fiat currency code. Currently, can be "USD", "EUR", "RUB", "BYN", "UAH", "GBP", "CNY", "KZT", "UZS", "GEL", "TRY", "AMD", "THB", "INR", "BRL", "IDR", "AZN", "AED", "PLN" and "ILS".
- **rate** (String)
The current rate of the *source* asset valued in the *target* currency.

AppStats

- **volume** (Number)
Total volume of paid invoices in USD.
- **conversion** (Number)
Conversion of all created invoices.
- **unique_users_count** (Number)
The unique number of users who have paid the invoice.
- **created_invoice_count** (Number)
Total created invoice count.
- **paid_invoice_count** (Number)
Total paid invoice count.
- **start_at** (String)
The date on which the statistics calculation was *started* in ISO 8601 format.

- **end_at** (String)

The date on which the statistics calculation was *ended* in ISO 8601 format.


Webhooks

Use Webhooks to get updates for your app, we will send an HTTPS POST request to the specified URL, containing a JSON-serialized [Update](#). In case of an unsuccessful request, we will give up after a reasonable amount of attempts.

Webhook request may be sent **at least one time**.

To make sure that the Webhook request was sent by Crypto Pay API, use a secret path in the URL, e.g. `https://www.example.com/<token>`. Since nobody else knows your app's token, you can be pretty sure it's us.

How to enable Webhooks?

Open [@CryptoBot](#) ([@CryptoTestnetBot](#) for testnet), go to [Crypto Pay](#) → *My Apps*, choose an app, then choose **Webhooks...** and tap  **Enable Webhooks**. Then enter HTTPS url which should be used by Crypto Pay API to post to.

Webhook updates

All requests sent by Crypto Pay API has this JSON object:

- **update_id** (Number)
Non-unique update ID.
- **update_type** (String)
Webhook update type. Supported update types:
`invoice_paid` – the update sent when the invoice is paid.
- **request_date** (String)
Date the request was sent in ISO 8601 format.
- **payload** ([Invoice](#))
Payload contains the class [Invoice](#).

Verifying webhook updates

You can verify the received update and the integrity of the received data by comparing the header parameter `crypto-pay-api-signature` and the hexadecimal representation of [HMAC-SHA-256](#) signature used to sign the entire request body (unparsed JSON string) with a secret key that is [SHA256](#) hash of [your app's token](#).

The full check might look like this:

```
const { createHash, createHmac } = require('crypto')
```

```
const checkSignature = (token, { body, headers }) => {
  const secret = createHash('sha256').update(token).digest()
  const checkString = JSON.stringify(body)
  const hmac = createHmac('sha256', secret).update(checkString).digest('hex')
  return hmac === signature['crypto-pay-api-signature']
}

console.log(checkSignature('XXX:XXXXX', { body: req.body, headers: req.headers}))
```

To prevent outdated data from being used, you can additionally check the `request_date` field which contains the date the request was sent in ISO 8601 format.

Libraries

This section lists some libraries and frameworks developed by the **Crypto Bot community** – you should take care to report any bugs you may find to the respective developers, as these projects are not maintained by Crypto Bot.

Ping us in Crypto Pay Developers chat if you would like your library to appear in this section.

Node.js

- [crypto-bot-api](#)

Python

- [aiocryptopay](#)

.NET

- [CryptoPay](#)

PHP

- [crypto-pay-api](#)

Go

- [cryptobot-sdk-golang](#)