

Pies (Deprecated) / Delete pie

# Trading212 Public API (v0)

Copy



Welcome to the official API reference for the Trading 212 Public API! This guide provides all the information you need to start building your own trading applications and integrations.

## General Information

This API is currently in **beta** and is under active development. We're continuously adding new features and improvements, and we welcome your feedback.

## Only for Invest and Stocks ISA

The API described here is enabled and usable only for **Invest and Stocks ISA** account types.

## API Environments

We provide two distinct environments for development and trading:

- **Paper Trading (Demo):** <https://demo.trading212.com/api/v0>
- **Live Trading (Real Money):** <https://live.trading212.com/api/v0>

You can test your applications extensively in the paper trading environment without risking real funds before moving to live trading.



## API Limitations

Please be aware of the following limitations for any order placement:

- **Supported account types:** The Trading 212 Public API is enabled and usable only for **Invest and Stocks ISA** account types.
- **Live trading order types:** For the live (real money) environment, only **Market Orders** are supported for execution via the API.
- **Order execution:** Orders can be executed only in the **primary account currency**
- **Multi-currency:** Multi-currency accounts are not currently supported through the API. Meaning your account, position and result values in the responses will be in the primary account currency.

## Key Concepts

- **Authentication:** Every request to the API must be authenticated using a secure key pair. See **Authentication** section below for details.

Ask AI

- **Rate Limiting:** All API calls are subject to rate limits to ensure fair usage and stability. See the **Rate Limiting** section for a full explanation.
- **IP Restrictions:** For enhanced security, you can optionally restrict your API keys to a specific set of IP addresses from within your Trading 212 account settings.
- **Selling Orders:** To execute a sell order, you must provide a **negative** value for the `quantity` parameter (e.g., `-10.5`). This is a core convention of the API.

---

## Quickstart

This simple example shows you how to retrieve your account's cash balance.

First, you must generate your API keys from within the Trading 212 app. For detailed instructions, please visit our Help Centre:

- [How to get your Trading 212 API key](#)

Once you have your **API Key** and **API Secret**, you can make your first call using `cURL` :

```
# Step 1: Replace with your actual credentials and Base64-encode them.

# The `-n` is important as it prevents adding a newline character.

CREDENTIALS=$(echo -n "<YOUR_API_KEY>:<YOUR_API_SECRET>" | base64)

# Step 2: Make the API call to the live environment using the encoded
credentials.

curl -X GET "https://live.trading212.com/api/v0/equity/account/cash" \
-H "Authorization: Basic $CREDENTIALS"
```

---

## Authentication

The API uses a secure key pair for authentication on every request. You must provide your **API Key** as the username and your **API Secret** as the password, formatted as an HTTP Basic Authentication header.

The `Authorization` header is constructed by Base64-encoding your `API_KEY:API_SECRET` string and prepending it with `Basic` .

### Building the Authorization Header

Here are examples of how to generate the required value in different environments.

**Linux or macOS (Terminal)**

✦ Ask AI

You can use the `echo` and `base64` commands. Remember to use the `-n` flag with `echo` to prevent it from adding a trailing newline, which would invalidate the credential string.

```
# This command outputs the required Base64-encoded string for your header.

echo -n "<YOUR_API_KEY>:<YOUR_API_SECRET>" | base64
```

## Python

This simple snippet shows how to generate the full header value.

```
import base64

# 1. Your credentials

api_key = "<YOUR_API_KEY>"

api_secret = "<YOUR_API_SECRET>"

# 2. Combine them into a single string

credentials_string = f"{api_key}:{api_secret}"

# 3. Encode the string to bytes, then Base64 encode it

encoded_credentials =
base64.b64encode(credentials_string.encode('utf-8')).decode('utf-8')

# 4. The final header value

auth_header = f"Basic {encoded_credentials}"

print(auth_header)
```

# Rate Limiting

To ensure high performance and fair access for all users, all API endpoints are subject to rate limiting.

**IMPORTANT NOTE:** All rate limits are applied on a per-account basis, regardless of which API key is used, and which IP address the request originates from.

✦ Ask AI

Specific rate limits are detailed in the reference for each endpoint.

## Response Headers

Every API response includes the following headers to help you manage your request frequency and avoid hitting limits.

- `x-ratelimit-limit` : The total number of requests allowed in the current time period.
- `x-ratelimit-period` : The duration of the time period in seconds.
- `x-ratelimit-remaining` : The number of requests you have left in the current period.
- `x-ratelimit-reset` : A Unix timestamp indicating the exact time when the limit will be fully reset.
- `x-ratelimit-used` : The number of requests you have already made in the current period.

## How It Works

The rate limiter allows for requests to be made in bursts. For example, an endpoint with a limit of `50 requests per 1 minute` does **not** strictly mean you can only make one request every 1.2 seconds. Instead, you could:

- Make a burst of all 50 requests in the first 5 seconds of a minute. You would then need to wait for the reset time indicated by the `x-ratelimit-reset` header before making more requests.
- Pace your requests evenly, for example, by making one call every 1.2 seconds, ensuring you always stay within the limit.

## Function-Specific Limits

In addition to the general rate limits on HTTP calls, some actions have their own functional limits. For example, there is a maximum of **50 pending orders** allowed per ticker, per account.

## Pagination

All list endpoints in the API that return a collection of items (such as historical orders, dividends, and transactions) use **cursor-based pagination** to handle large data sets.

## Parameters

- `limit` (integer): Specifies the maximum number of items to return in a single request.
  - **Default:** 20
  - **Maximum:** 50
- `cursor` (string | number): A pointer to a specific item in the dataset. This tells the API where to start the next page of results.

## How to Paginate

The easiest way to paginate is by using the `nextPagePath` field returned in the response.

✦ Ask AI

1. Make your initial request to a list endpoint (e.g., `/api/v0/equity/history/orders` ) with an optional `limit` parameter. Do not include a `cursor` .
2. The API will return a response object. This object will contain a list of `items` and a `nextPagePath` field.
3. If the `nextPagePath` field is `null` , you have reached the end of the data, and there are no more pages.
4. If `nextPagePath` is not `null` , **use the entire string value of** `nextPagePath` as the path for your next request. This string contains all the necessary parameters (like `limit` and `cursor` ) to get the next page.
5. Repeat this process until `nextPagePath` is `null` .

## Example

Here is a step-by-step example of fetching all transactions, 2 at a time.

### Request 1: Get the first page

```
curl -X GET "https://demo.trading212.com/api/v0/equity/history/orders?limit=2" \
-u "API_KEY:API_SECRET"
```

### Response 1: Note the nextPagePath

```
{
  "items": [
    { "id": 987654321, "ticker": "AAPL_US_EQ", ... },
    { "id": 987654320, "ticker": "MSFT_US_EQ", ... }
  ],
  "nextPagePath": "/api/v0/equity/history/orders?limit=2&cursor=1760346100000"
}
```

### Request 2: Use the full nextPagePath for the next request

```
curl -X GET "https://demo.trading212.com/api/v0/equity/history/orders?limit=2&cursor=1760346100000" \
-u "API_KEY:API_SECRET"
```

### Response 2: Get the next page (and a new nextPagePath)

```
{
  "items": [
    { "id": 987654319, "ticker": "AAPL_US_EQ", ... },
    { "id": 987654320, "987654318": "MSFT_US_EQ", ... }
  ],
  "nextPagePath": "/api/v0/equity/history/orders?limit=2&cursor=1660015723000"
}
```

### Request 3: Get the final page

```
curl -X GET "https://demo.trading212.com/api/v0/equity/history/orders?limit=2&cursor=1660015723000" \
-u "API_KEY:API_SECRET"
```

Response 3: `nextPagePath` is `null`, indicating the end

```
{
  "items": [
    { "id": 987654319, "ticker": "AAPL_US_EQ", ... },
    { "id": 987654320, "987654318": "MSFT_US_EQ", ... }
  ],
  "nextPagePath": null
}
```

✦ Ask AI

# Useful Links

Here are some additional resources that you may find helpful.

- [Trading 212 API Terms](#)
- [Trading 212 Community Forum](#) - A great place to ask questions and share what you've built.

Download OpenAPI description

JSON

api.json


↓

api.yaml


↓

Languages


curl




Go



Python



Node.js



Java

Servers

https://demo.trading212.com/

⌂

https://live.trading212.com/

⌂

# Accounts

Access fundamental information about your trading account. Retrieve details such as your account and current cash balance.

Copy

⌵


Ask AI

### Operations

```
GET /api/v0/equity/account/summary
```

+ Show

## Instruments

 Copy



Discover what you can trade. These endpoints provide comprehensive lists of all tradable instruments and the exchanges they belong to, including details like tickers and trading hours.


### Operations

```
GET /api/v0/equity/metadata/exchanges
```

```
GET /api/v0/equity/metadata/instruments
```

+ Show

## Orders

 Copy



### Order Limitations

- Orders can be executed only in the **main account currency**
- Only **Market Orders** are supported for the live (real money) environment

Place, monitor, and cancel equity trade orders. This section provides the core functionality for programmatically executing your trading strategies for stocks and ETFs.

### Operations


```
GET /api/v0/equity/orders
```

 Ask AI

POST	/api/v0/equity/orders/limit
POST	/api/v0/equity/orders/market
POST	/api/v0/equity/orders/stop
POST	/api/v0/equity/orders/stop_limit
DELETE	/api/v0/equity/orders/{id}
GET	/api/v0/equity/orders/{id}

+ Show

## Positions

 Copy



Get a real-time overview of all your open positions, including quantity, average price, and current profit or loss.

### Operations

GET	/api/v0/equity/positions
-----	--------------------------

+ Show

## Historical events

 Copy



Review your account's trading history. Access detailed records of past orders, dividend payments, and cash transactions, or generate downloadable CSV reports for analysis and record-keeping.

### Operations

GET	/api/v0/equity/history/dividends
GET	/api/v0/equity/history/exports



 Ask AI



<code>POST /api/v0/equity/history/exports</code>
<code>GET /api/v0/equity/history/orders</code>
<code>GET /api/v0/equity/history/transactions</code>

+ Show

## Pies (Deprecated)

 Copy 

Manage your investment Pies. Use these endpoints to create, view, update, and delete your custom portfolios, making automated and diversified investing simple.

**Deprecation notice:** The current state of the Pies API, while still operational, won't be further supported and updated.


### Operations

<code>GET /api/v0/equity/pies</code> <span>Deprecated</span>
<code>POST /api/v0/equity/pies</code> <span>Deprecated</span>
<code>DELETE /api/v0/equity/pies/{id}</code> <span>Deprecated</span>
<code>GET /api/v0/equity/pies/{id}</code> <span>Deprecated</span>
<code>POST /api/v0/equity/pies/{id}</code> <span>Deprecated</span>
<code>POST /api/v0/equity/pies/{id}/duplicate</code> <span>Deprecated</span>

## Fetch all pies Deprecated

 Copy 

### Request

 Ask AI

Fetches all pies for the account

Rate limit: 1 req / 30s

 Security

View security details

authWithSecretKey and legacyApiKeyHeader

GET /api/v0/equity/pies

Python

```
import requests

url = "https://demo.trading212.com/api/v0/equity/pies"

headers = {"Authorization": "YOUR_API_KEY_HERE"}

response = requests.get(url, headers=headers, auth=('<username>', '<password>'))

data = response.json()
print(data)
```

Try it

Responses 200 401 403 408 429

OK

Body application/json

Array [

cash number

Amount of money put into the pie in account currency

dividendDetails object, (DividendDetails)

+ Show 3 properties

id integer, (int64)


progress number

Progress of the pie based on the set goal

Example: 0.5

result object, (InvestmentResult)

Ask AI

 Show 4 properties

**status** string

Status of the pie based on the set goal

Enum "AHEAD" "ON\_TRACK" "BEHIND"

1

Response

200401403408429

application/json

[

✓ {

"cash": 0,

> "dividendDetails": { ... },

"id": 0,





"progress": 0.5,

> "result": { ... },

"status": "AHEAD"

}

]



Was this helpful?  

# Create pie Deprecated

 Copy




## Request


Expand all 

Creates a pie for the account by given params

**Rate limit:** 1 req / 5s

 Security

View

 Ask AI

authWithSecretKey and legacyApiKeyHeader

## Body application/json

required

**dividendCashAction** string

Enum "REINVEST" "TO\_ACCOUNT\_CASH"

**endDate** string, (date-time)


**goal** number

Total desired value of the pie in account currency


**icon** string

**instrumentShares** object

Example: `{"AAPL_US_EQ":0.5,"MSFT_US_EQ":0.5}`

 Show property

**name** string

**POST** /api/v0/equity/pies 

Python 

```
import requests

url = "https://demo.trading212.com/api/v0/equity/pies"

payload = {
    "dividendCashAction": "REINVEST",
    "endDate": "2019-08-24T14:15:22Z",
    "goal": 0,
    "icon": "string",
    "instrumentShares": {
        "AAPL_US_EQ": 0.5,
        "MSFT_US_EQ": 0.5
    },
    "name": "string"
}

headers = {
    "Content-Type": "application/json",
    "Authorization": "YOUR_API_KEY_HERE"
}

response = requests.post(url, json=payload, headers=headers, auth=('<username>', '<password>'))

data = response.json()
print(data)
```

[Try it](#)

## Responses

☐ 200 ☐ 400 ☐ 401 ☐ 403 ☐ 408 ☐ 429

[Expand all](#)

OK

### Body application/json

**instruments** Array of objects, (AccountBucketInstrumentResult)

[+ Show 6 array properties](#)

**settings** object, (AccountBucketDetailedResponse)

[+ Show 10 properties](#)


### Response

☐ 200 ☐ 400 ☐ 401 ☐ 403 ☐ 408 ☐ 429

application/json

[Ask AI](#)

```
{
  < "instruments": [
    > { ... }
  ],
  < "settings": {
    "creationDate": "2019-08-24T14:15:22Z",
    "dividendCashAction": "REINVEST",
    "endDate": "2019-08-24T14:15:22Z",
    "goal": 0,
    "icon": "string",
    "id": 0,
    "initialInvestment": 0,
    > "instrumentShares": { ... },
    "name": "string",
    "publicUrl": "string"
  }
}
```

Was this helpful?  

# Delete pie Deprecated

 Copy



## Request

Deletes a pie by given id

**Rate limit:** 1 req / 5s

 Security View security details

authWithSecretKey and legacyApiKeyHeader

## Path

**id** integer, (int64) required

 Ask AI

**DELETE** /api/v0/equity/pies/{id}

Python

```
import requests

id = "YOUR_id_PARAMETER"
url = "https://demo.trading212.com/api/v0/equity/pies/" + id

headers = {"Authorization": "YOUR_API_KEY_HERE"}

response = requests.delete(url, headers=headers, auth=('<username>', '<password>'))

data = response.json()
print(data)
```

**Responses**   200   401   403   404   408   429

OK

Was this helpful?

# Fetch a pie

Deprecated

Copy

## Request

Fetches a pies for the account with detailed information

**Rate limit:** 1 req / 5s

**Security** View security details

authWithSecretKey and legacyApiKeyHeader 

**Ask AI**

Path

**id** integer, (int64) **required**

GET /api/v0/equity/pies/{id} ▾

Python ▾

```
import requests

id = "YOUR_id_PARAMETER"
url = "https://demo.trading212.com/api/v0/equity/pies/" + id

headers = {"Authorization": "YOUR_API_KEY_HERE"}

response = requests.get(url, headers=headers, auth=('<username>', '<password>'))

data = response.json()
print(data)
```

Try it

Responses 

200

401

403

408

429

Expand all ↗

OK

Body application/json

**instruments** Array of objects, (AccountBucketInstrumentResult)

+

 Show 6 array properties

**settings** object, (AccountBucketDetailedResponse)

+

 Show 10 properties

Response

200

401

403

408

429

application/json

Ask AI



```
{
  < "instruments": [
    > { ... }
  ],
  < "settings": {
    "creationDate": "2019-08-24T14:15:22Z",
    "dividendCashAction": "REINVEST",
    "endDate": "2019-08-24T14:15:22Z",
    "goal": 0,
    "icon": "string",
    "id": 0,
    "initialInvestment": 0,
    > "instrumentShares": { ... },
    "name": "string",
    "publicUrl": "string"
  }
}
```


Was this helpful?  

# Update pie Deprecated

 Copy



## Request

Expand all 

Updates a pie for the account by given params

**Rate limit:** 1 req / 5s

 Security View security details

authWithSecretKey and legacyApiKeyHeader

## Path

**id** integer, (int64) required

**Body** application/json  
required

 Ask AI

**dividendCashAction** string

Enum "REINVEST" "TO\_ACCOUNT\_CASH"

**endDate** string, (date-time)


**goal** number

Total desired value of the pie in account currency


**icon** string

**instrumentShares** object

Example: {"AAPL\_US\_EQ":0.5,"MSFT\_US\_EQ":0.5}

 Show property

**name** string

**POST** /api/v0/equity/pies/{id} 

Python 

```
import requests

id = "YOUR_id_PARAMETER"
url = "https://demo.trading212.com/api/v0/equity/pies/" + id

payload = {
    "dividendCashAction": "REINVEST",
    "endDate": "2019-08-24T14:15:22Z",
    "goal": 0,
    "icon": "string",
    "instrumentShares": {
        "AAPL_US_EQ": 0.5,
        "MSFT_US_EQ": 0.5
    },
    "name": "string"
}

headers = {
    "Content-Type": "application/json",
    "Authorization": "YOUR_API_KEY_HERE"
}

response = requests.post(url, json=payload, headers=headers, auth=('<username>', '<password>'))

data = response.json()
print(data)
```

Try it

Responses

200400401403408429

Expand all ↗

OK

Body application/json

**instruments** Array of objects, (AccountBucketInstrumentResult)

+

 Show 6 array properties

**settings** object, (AccountBucketDetailedResponse)

+

 Show 10 properties

Response

200400401403408429

application/json

↗

↶

↵

⌵

"instruments": [

>

{ ... }

],

⌵

"settings": {

"creationDate": "2019-08-24T14:15:22Z",

"dividendCashAction": "REINVEST",

"endDate": "2019-08-24T14:15:22Z",

"goal": 0,

"icon": "string",

"id": 0,

"initialInvestment": 0,

>

"instrumentShares": { ... },

"name": "string",

"publicUrl": "string"

}

}

Was this helpful? 


👍


👎

✦

 Ask AI

# Duplicate pie Deprecated

 Copy



## Request

Duplicates a pie for the account

**Rate limit:** 1 req / 5s

 Security

[View security details](#)

authWithSecretKey and legacyApiKeyHeader

## Path

**id** integer, (int64) required

**Body** application/json  
required

**icon** string

**name** string

POST /api/v0/equity/pies/{id}/duplicate 

Python 

```
import requests


id = "YOUR_id_PARAMETER"
url = "https://demo.trading212.com/api/v0/equity/pies/" + id + "/duplicate"

payload = {
    "icon": "string",
    "name": "string"
}

headers = {
    "Content-Type": "application/json",
    "Authorization": "YOUR_API_KEY_HERE"
}


response = requests.post(url, json=payload, headers=headers, auth=('<username>', '<password>'))

data = response.json()
print(data)
```

 Try it

## Responses


☐ 200 ☐ 400 ☐ 401 ☐ 403 ☐ 408 ☐ 429

Expand all 


OK

### Body application/json

**instruments** Array of objects, (AccountBucketInstrumentResult)

 Show 6 array properties

**settings** object, (AccountBucketDetailedResponse)

 Show 10 properties

### Response

☐ 200 ☐ 400 ☐ 401 ☐ 403 ☐ 408 ☐ 429

application/json

 Ask AI

```
{
  ✓ "instruments": [
    > { ... }
  ],
  ✓ "settings": {
    "creationDate": "2019-08-24T14:15:22Z",
    "dividendCashAction": "REINVEST",
    "endDate": "2019-08-24T14:15:22Z",
    "goal": 0,
    "icon": "string",
    "id": 0,
    "initialInvestment": 0,
    > "instrumentShares": { ... },
    "name": "string",
    "publicUrl": "string"
  }
}
```

Was this helpful?  

✦ Ask AI