Brickfront Documentation

Release 0.0.4

Callum Bartlett

Dec 31, 2017

Contents:

CHAPTER 1

Intro

Brickfront is a basic interface for working through Brickset's API. It should be pretty simple to use, though I do have the code reference if there's anyhing too difficult.

Installation:

```
pip install brickfront
```

CHAPTER 2

Getting Started

There's quite basic usage. For most things you don't need an API key, but for others you may need to get one.

First you need to make a *Client* object.

```
>>> import brickfront
>>> client = brickfront.Client(API_KEY)
```

From there, you can make requests through your client to be able to get different sets.

```
>>> setList = client.getSets(query='star wars')
>>> len(setList)
20
>>> build = setList[18]
>>> build.name
"Jabba's Palace"
>>> build.setID
'2141'
>>> build.pieces
231
>>> build.priceUK
'27.99'
```

Most code is fully internally documented, so it will autofill and properly interface with Python's *help* function.

CHAPTER 3

Code Reference

3.1 Client

class brickfront.client.**Client**(*apiKey*, *raiseError=True*)

A frontend authenticator for Brickset.com's API. All endpoints require an API key.

Parameters

- **apiKey** (*str*) – The API key you got from Brickset.
- **raiseError** (*bool*) – (optional) Whether or not you want an error to be raised on an invalid API key.

Raises ***brickfront.errors.InvalidApiKey*** – If the key provided is invalid.

checkKey(*key=None*)

Checks that an API key is valid.

Parameters **key** (*str*) – (optional) A key that you want to check the validity of. Defaults to the one provided on initialization.

Returns If the key is valid, this method will return `True`.

Return type *bool*

Raises *brickfront.errors.InvalidApiKey*

static checkResponse(*request*)

Returns if a request has an okay error code, otherwise raises InvalidRequest.

getAdditionalImages(*setID*)

Gets a list of URLs containing images of the set.

Parameters **setID** (*str*) – The ID of the set you want to grab the images for.

Returns A list of URL strings.

Return type list

Warning: An empty list will be returned if there are no additional images, or if the set ID is invalid.

`getInstructions`(*setID*)

Get the instructions for a set.

Parameters **`setID`** (*`str`*) – The ID for the set you want to get the instructions of.

Returns A list of URLs to instructions.

Return type List[*dict*]

Warning: An empty list will be returned if there are no instructions, or if the set ID is invalid.

`getRecentlyUpdatedSets`(*minutesAgo*)

Gets the information of recently updated sets.

Parameters **`minutesAgo`** (*`int`*) – The amount of time ago that the set was updated.

Returns A list of Build instances that were updated within the given time.

Return type list

Warning: An empty list will be returned if there are no sets in the given time limit.

`getReviews`(*setID*)

Get the reviews for a set.

Parameters **`setID`** (*`str`*) – The ID of the set you want to get the reviews of.

Returns A list of reviews.

Return type List[*`brickfront.review.Review`*]

Warning: An empty list will be returned if there are no reviews, or if the set ID is invalid.

`getSet`(*setID*)

Gets the information of one specific build using its Brickset set ID.

Parameters **`setID`** (*`str`*) – The ID of the build from Brickset.

Returns A single Build object.

Return type *`brickfront.build.Build`*

Raises **`brickfront.errors.InvalidSetID`** – If no sets exist by that ID.

`getSets`(***kwargs*)

A way to get different sets from a query. All parameters are optional, but you should probably use some (so that you get results)

Parameters

- **`query`** (*`str`*) – The thing you're searching for.
- **`theme`** (*`str`*) – The theme of the set.
- **`subtheme`** (*`str`*) – The subtheme of the set.

- **setNumber** (*str*) – The LEGO set number.
- **year** (*str*) – The year in which the set came out.
- **owned** (*int*) – Whether or not you own the set. Only works when logged in with `login()`. Set to *1* to make true.
- **wanted** (*int*) – Whether or not you want the set. Only works when logged in with `login()`. Set to *1* to make true.
- **orderBy** (*str*) – How you want the set ordered. Accepts 'Number', 'YearFrom', 'Pieces', 'Minifigs', 'Rating', 'UKRetailPrice', 'USRetailPrice', 'CARetailPrice', 'EU-RetailPrice', 'Theme', 'Subtheme', 'Name', 'Random'. Add 'DESC' to the end to sort descending, e.g. NameDESC. Case insensitive. Defaults to 'Number'.
- **pageSize** (*int*) – How many results are on a page. Defaults to 20.
- **pageNumber** (*int*) – The number of the page you're looking at. Defaults to 1.
- **userName** (*str*) – The name of a user whose sets you want to search.

Returns A list of `brickfront.build.Build` objects.

Return type list

login (*username*, *password*)

Logs into Brickset as a user, returning a userhash, which can be used in other methods. The user hash is stored inside the client (`userHash`).

Parameters

- **username** (*str*) – Your Brickset username.
- **password** (*str*) – Your Brickset password.

Returns If the login is valid, this will return `True`.

Return type *bool*

Raises `brickfront.errors.InvalidLoginCredentials`

3.2 Build

class brickfront.build.**Build** (*data*, *client*)

A class holding the information of a LEGO set. Some attributes may be `None`. There is no need to create an instance of a `Build` object yourself - it won't go well.

Variables

- **setID** (*int*) – The set ID, as used on Brickset.
- **number** (*str*) – The LEGO ID number of the set.
- **variant** (*int*) – The variant of the set, as used on Brickset.
- **name** (*str*) – The name of the set.
- **year** (*str*) – The year the set was released.
- **theme** (*str*) – The theme of the set.
- **themeGroup** (*str*) – The theme group, as used on /browse/sets
- **subtheme** (*str*) – The subtheme of the set.

- **pieces** (*int*) – How many pieces the set has.
- **minifigs** (*int*) – The amount of minifigs what come with the set.
- **imageURL** (*str*) – A URL to an image of the set.
- **bricksetURL** (*str*) – The URL to the page for the set on Brickset.
- **released** (*bool*) – Whether or not the set is released.
- **owned** (*bool*) – Whether or not own the set. Only works when logged in.
- **wanted** (*bool*) – Whether or not you want the set. Only works when logged in.
- **quantityOwned** (*int*) – The number of these that you own. Only works when logged in.
- **ACMDataCount** (*int*) – Number of ACM records you have modified for this set.
- **userNotes** (*str*) – The notes that you have on the product. Only works when logged in.
- **ownedByTotal** (*int*) – The number of people who own this set.
- **wantedByTotal** (*int*) – The number of people who want this set.
- **priceUK** (*str*) – The price of the set.
- **priceUS** (*str*) – The price of the set.
- **priceCA** (*str*) – The price of the set.
- **priceEU** (*str*) – The price of the set.
- **dateAddedToStore** (*datetime.datetime*) – The date that the set was added to the US LEGO store website.
- **dateRemovedFromStore** (*datetime.datetime*) – The date that the set was removed from the US LEGO store website. If `None`, then the set is still available.
- **rating** (*float*) – The overall rating of the set on Brickset.
- **reviewCount** (*int*) – How many reviews the set has on Brickset.
- **packagingType** (*str*) – How the set is packaged
- **availability** (*str*) – Where the set is available.
- **instructionsCount** (*int*) – How many sets of instructions the set has.
- **additionalImageCount** (*int*) – How many additional images the set has.
- **EAN** (*str*) – The standard barcode number of the set.
- **UPC** (*str*) – The standard barcode number of the set.
- **description** (*str*) – The description of the set.
- **lastUpdated** (*datetime.datetime*) – When the set was last updated on Brickset.
- **additionalImages** (*list*) – The URLs to the additional images of the set.
- **reviews** (*list*) – A list of *brickfront.review.Review* objects for the reviews of the set.

getAdditionalImages()

The same as calling `client.getAdditionalImages(build.setID)`.

Returns A list of URL strings.

Return type list

getInstructions()
The same as calling `client.getInstructions(build.setID)`

Returns A list of instructions.

Return type list

getReviews()
The same as calling `client.getReviews(build.setID)`.

Returns A list of `brickfront.review.Review` objects.

Return type list

3.3 Review

class brickfront.review.**Review**(*data*)
A class holding data for a review.

Variables

- **author** – The *str* name of the person who wrote the review.
- **datePosted** – The *str* date that the review was posted.
- **overallRating** – The *int* rating that was given.
- **parts** – *int*
- **buildingExperience** – *int*
- **playability** – *int*
- **valueForMoney** – *int*
- **title** – *str*
- **review** – *str*
- **HTML** – *bool*

3.4 Exceptions

exception brickfront.errors.**InvalidApiKey**
The API key provided was invalid and cannot be used.

exception brickfront.errors.**InvalidLoginCredentials**
The login credentials used were invalid.

exception brickfront.errors.**InvalidRequest**
The request that was sent to the server was invalid.

exception brickfront.errors.**InvalidSetID**
The set ID that was passed is invalid - eg it doesn't exist.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index