dblpy Documentation

Release 0.3.0

Francis Taylor

May 19, 2019

Contents

CHAPTER 1

What's New

This page keeps a detailed human friendly rendering of what's new and changed in specific versions.

1.1 v0.3.0

- *Client* now has `autopost` kwarg that will post server count automatically every 30 minutes
- Fixed code 403 errors
- Added `on_dbl_vote`, an event that is called when you test your webhook
- Added `on_dbl_test`, an event that is called when someone tests your webhook

1.2 v0.2.1

- Added webhook
- Removed support for discord.py versions lower than 1.0.0
- Made *Client.get_weekend_status()* return a boolean value
- Added webhook example in README
- Removed `post_server_count` and `get_server_count`

1.3 v0.2

- Added `post_guild_count`
- Made `post_server_count` an alias for `post_guild_count`
- Added `get_guild_count`

- Made `get_server_count` an alias for `get_guild_count`
- Added *`Client.get_weekend_status()`*
- Removed all parameters from *`Client.get_upvote_info()`*
- Added limit to *`Client.get_bots()`*
- Fixed example in README

1.4 v0.1.6

- Bug fixes & improvements

1.5 v0.1.4

- Initial ratelimit handling

1.6 v0.1.3

- Added documentation
- Fixed some minor bugs

1.7 v0.1.2

Initial release

- Working
  - POSTing server count
  - GET bot info, server count, upvote count, upvote info
  - GET all bots
  - GET specific user info
  - GET widgets (large and small) including custom ones. See discordbots.org/api/docs for more info.
- Not Working / Implemented
  - Searching for bots via the api

CHAPTER 2

API Reference

The following section outlines the API of dblpy.

2.1 Version Related Info

There are two main ways to query version information about the library.

`dbl.`**`version_info`**
: A named tuple that is similar to sys.version_info.

  Just like sys.version_info the valid values for `releaselevel` are 'alpha', 'beta', 'candidate' and 'final'.

`dbl.`**`__version__`**
: A string representation of the version. e.g. `'0.1.0-alpha0'`.

2.2 Client

Note: All of the following functions return their data as a JSON object (except widget generation)!

class `dbl.`**Client**(*bot*, *token*, ***kwargs*)
: Represents a client connection that connects to discordbots.org. This class is used to interact with the DBL API.

  Parameters

  - **token** – An API Token
  - **bot** – An instance of a discord.py Bot or Client object
  - ****loop** (*Optional[event loop]*) – The event loop to use for asynchronous operations. Defaults to `bot.loop`.
  - ****session** (*Optional*) – The aiohttp session to use for requests to the API.

- ****webhook_auth** (*Optional*) – The string for Authorization you set on the site for verification.
- ****webhook_path** (*Optional*) – The path for the webhook request.
- ****webhook_port** (*Optional*) – The port to run the webhook on. Will activate webhook when set.

guild_count()
Gets the guild count from the Client/Bot object

close()
This function is a coroutine.

Closes all connections.

generate_widget_large(*bot_id: int = None*, *top: str = '2C2F33'*, *mid: str = '23272A'*, *user: str = 'FFFFFF'*, *cert: str = 'FFFFFF'*, *data: str = 'FFFFFF'*, *label: str = '99AAB5'*, *highlight: str = '2C2F33'*)
This function is a coroutine.

Generates a custom large widget. Do not add # to the color codes (e.g. #FF00FF become FF00FF).

Parameters

- **bot_id** (*int*) – The bot_id of the bot you wish to make a widget for.
- **top** (*str*) – The hex color code of the top bar.
- **mid** (*str*) – The hex color code of the main section.
- **user** (*str*) – The hex color code of the username text.
- **cert** (*str*) – The hex color code of the certified text (if applicable).
- **data** (*str*) – The hex color code of the statistics (numbers only e.g. 44) of the bot.
- **label** (*str*) – The hex color code of the description (text e.g. guild count) of the statistics.
- **highlight** (*str*) – The hex color code of the data boxes.

Returns URL of the widget

Return type str

generate_widget_small(*bot_id: int = None*, *avabg: str = '2C2F33'*, *lcol: str = '23272A'*, *rcol: str = '2C2F33'*, *ltxt: str = 'FFFFFF'*, *rtxt: str = 'FFFFFF'*)
This function is a coroutine.

Generates a custom large widget. Do not add # to the color codes (e.g. #FF00FF become FF00FF).

Parameters

- **bot_id** (*int*) – The bot_id of the bot you wish to make a widget for.
- **avabg** (*str*) – The hex color code of the background of the avatar (if the avatar has transparency).
- **lcol** (*str*) – The hex color code of the left background color.
- **rcol** (*str*) – The hex color code of the right background color.
- **ltxt** (*str*) – The hex color code of the left text.
- **rtxt** (*str*) – The hex color code of the right text.

Returns URL of the widget

Return type str

get_bot_info(*bot_id: int = None*)
This function is a coroutine.

Gets information about a bot from discordbots.org

Parameters **bot_id** (*int[Optional]*) – The bot_id of the bot you want to lookup.

Returns **bot_info** – Information on the bot you looked up. https://discordbots.org/api/docs#bots

Return type dict

get_bots(*limit: int = 50*, *offset: int = 0*)
This function is a coroutine.

Gets information about listed bots on discordbots.org

Parameters

- **limit** (*int[Optional]*) – The number of results you wish to lookup. Defaults to 50. Max 500.
- **offset** (*int[Optional]*) – The amount of bots to skip. Defaults to 0.

Returns **bots** – Returns info on the bots on DBL. https://discordbots.org/api/docs#bots

Return type dict

get_guild_count(*bot_id: int = None*)
This function is a coroutine.

Gets a guild count from discordbots.org

Parameters **bot_id** (*int[Optional]*) – The bot_id of the bot you want to lookup. Defaults to the Bot provided in Client init

Returns **stats** – The guild count and shards of a bot. The date object is returned in a datetime.datetime object

Return type dict

get_upvote_info(*bot_id*)
This function is a coroutine.

Gets information about who upvoted a bot from discordbots.org

---

Note: This API endpoint is available to the owner of the bot only.

Returns **votes** – Info about who upvoted your bot.

Return type dict

get_user_info(*user_id: int*)
This function is a coroutine.

Gets information about a user on discordbots.org

Parameters **user_id** (*int*) – The user_id of the user you wish to lookup.

Returns **user_data** – Info about the user. https://discordbots.org/api/docs#users

Return type dict

get_weekend_status()
This function is a coroutine.

Gets weekend status from discordbots.org

Returns **weekend status** – The boolean value of weekend status.

Return type bool

get_widget_large(*bot_id: int = None*)
This function is a coroutine.

Generates the default large widget.

Parameters **bot_id** (*int*) – The bot_id of the bot you wish to make a widget for.

Returns **URL of the widget**

Return type str

get_widget_small(*bot_id: int = None*)
This function is a coroutine.

Generates the default small widget.

Parameters **bot_id** (*int*) – The bot_id of the bot you wish to make a widget for.

Returns **URL of the widget**

Return type str

post_guild_count(*shard_count: int = None*, *shard_no: int = None*)
This function is a coroutine.

Posts the guild count to discordbots.org

Parameters

- **shard_count** (*int[Optional]*) – The total number of shards.
- **shard_no** (*int[Optional]*) – The index of the current shard. DBL uses 0 based indexing for shards.

2.3 Event reference

dbl.**on_dbl_vote**(*data*)
Called when someone votes for your bot on discordbots.org

Parameters **data** – The data with vote info returned in dict object

Example:

```
@bot.event
async def on_dbl_vote(data):
    print(data)

# Will output the following:
# {
# 'type': "upvote",
# 'user': "247741991310327810",
# 'bot': "264811613708746752",
# 'isWeekend': False
# }
```

dbl.**on_dbl_test**(*data*)
Called when someone tests webhook system for your bot on discordbots.org

Parameters **data** – The data with test info returned in dict object

Example:

```
@bot.event
async def on_dbl_test(data):
    print(data)

# Will output the following:
# {
# 'type': "type",
# 'user': "247741991310327810",
# 'bot': "264811613708746752",
# 'isWeekend': True
# }
```

2.4 Exceptions

The following exceptions are thrown by the library.

exception dbl.**DBLException**
Base exception class for dblpy

Ideally speaking, this could be caught to handle any exceptions thrown from this library.

exception dbl.**UnauthorizedDetected**
Exception that's thrown when no API Token is provided

Subclass of *DBLException*

exception dbl.**ClientException**
Exception that's thrown when an operation in the *Client* fails.

These are usually for exceptions that happened due to user input.

exception dbl.**HTTPException**(*response*, *message*)
Exception that's thrown when an HTTP request operation fails.

response
The response of the failed HTTP request. This is an instance of aiohttp.ClientResponse.

text
The text of the error. Could be an empty string.

Type str

exception dbl.**Unauthorized**(*response*, *message*)
Exception that's thrown for when status code 401 occurs.

Subclass of *HTTPException*

exception dbl.**Forbidden**(*response*, *message*)
Exception that's thrown for when status code 403 occurs.

Subclass of *HTTPException*

exception dbl.**NotFound**(*response*, *message*)
Exception that's thrown for when status code 404 occurs.

Subclass of *HTTPException*

exception dbl.**InvalidArgument**
Exception that's thrown when an argument to a function is invalid some way (e.g. wrong value or wrong type).

This could be considered the analogous of `ValueError` and `TypeError` except derived from *ClientException* and thus *DBLException*.

exception dbl.**ConnectionClosed**(*original*)
Exception that's thrown when the gateway connection is closed for reasons that could not be handled internally.

code
The close code of the websocket.

Type int

reason
The reason provided for the closure.

Type str

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Index