
BoticordPY

Release 1.5

Grey Cat

Sep 24, 2021

CONTENTS:

- 1 Quickstart 3**
 - 1.1 Installation 3
 - 1.2 Examples 3
- 2 Main 5**
 - 2.1 BoticordClient 5
- 3 Modules 7**
 - 3.1 Bots 7
 - 3.2 Servers 7
 - 3.3 Users 8
- 4 BoticordWebhook 9**
- 5 Reference 11**
 - 5.1 Event Reference 11
 - 5.2 Types 12
- 6 Exceptions 13**
- 7 Links 15**
- Python Module Index 17**
- Index 19**

This is a documentation for simple python module to work with the boticord api.

QUICKSTART

1.1 Installation

Enter one of these commands to install the library:

```
pip install boticordpy
```

```
python3 -m pip install boticordpy
```

Or just clone the repo: <https://github.com/grey-cat-1908/boticordpy>

1.2 Examples

Without Using Cogs System

Post bot stats when bot is ready.

```
from discord.ext import commands

from boticordpy import BoticordClient

bot = commands.Bot(command_prefix="!")
boticord = BoticordClient(bot, "your-boticord-token")

@bot.event
async def on_ready():
    stats = {"servers": len(bot.guilds), "shards": bot.shard_count, "users": len(bot.
    ↪ users)}
    await boticord.Bots.postStats(stats)

bot.run("your-bot-token")
```

Using Cogs System

Cog with automatically stats post (every 15 minutes) + bot's owner command that can be used to post stats.

```
from discord.ext import commands
```

(continues on next page)

(continued from previous page)

```
from boticordpy import BoticordClient

class BoticordCog(commands.Cog):
    def __init__(self, bot):
        self.bot = bot
        self.boticord = BoticordClient(self.bot, "your-boticord-token")
        self.boticord.start_loop()

    @commands.command(name="boticord-update")
    @commands.is_owner()
    async def boticord_update(self, ctx):
        """
        This commands can be used by owner to post stats to boticord
        """
        stats = {"servers": len(self.bot.guilds), "shards": 0, "users": len(self.bot.
↪users)}
        await self.boticord.Bots.postStats(stats)

def setup(bot):
    bot.add_cog(BoticordCog(bot))
```


This class is used to make it much easier to use the Boticord API.

2.1 BoticordClient

class boticordpy.**BoticordClient**(*bot*, *token=None*, ***kwargs*)

This class is used to make it much easier to use the Boticord API.

Parameters

- **bot** (commands.Bot | commands.AutoShardedBot) – The discord.py Bot instance
- **token** (str) – boticord api key

Bots

modules.bots.Bots with all arguments filled.

Type *modules.bots.Bots*

Servers

modules.servers.Servers with all arguments filled.

Type *modules.servers.Servers*

Users

modules.users.Users with all arguments filled.

Type *modules.users.Users*

event(*event_name: str*)

A decorator that registers an event to listen to. You can find all the events on Event Reference page.

:param event_name str: boticord event name

start_loop(*sleep_time: Optional[int] = None*) → None

Can be used to post stats automatically.

Parameters **sleep_time** (int) – stats posting interval - can be not specified or None (default interval - 15 minutes)

MODULES

This page describes all the modules of boticordpy.

3.1 Bots

class boticordpy.modules.**Bots**(*bot*, ***kwargs*)

Class with methods to work with Boticord API Bots.

Parameters **bot** (`commands.Bot` | `commands.AutoShardedBot`) – The discord.py Bot instance

async **getBotComments**(*botID: int*)

Returns comments of the discord bot with the given ID.

Parameters **botID** (`int`) – Discord Bot's ID

async **getBotInfo**(*botID: int*)

Returns information about discord bot with the given ID.

Parameters **botID** (`int`) – Discord Bot's ID

async **postStats**(*stats: dict*)

Post stats to Boticord API.

Parameters **stats** (`dict`) – A dictionary of {`guilds: int`, `shards: int`, `users: int`}

3.2 Servers

class boticordpy.modules.**Servers**(*bot*, ***kwargs*)

Class with methods to work with Boticord API Servers.

Parameters **bot** (`commands.Bot` | `commands.AutoShardedBot`) – The discord.py Bot instance

async **getServerComments**(*serverID: int*)

Returns comments of the discord server with the given ID.

Parameters **serverID** (`int`) – Discord Server's ID

async **getServerInfo**(*serverID: int*)

Returns information about discord server with the given ID.

Parameters **serverID** (`int`) – Discord Server's ID

async **postServerStats**(*message: discord.message.Message*, *custom_stats: Optional[dict] = None*)

Post server stats to Boticord API.

Parameters

- **message** (discord.Message) – Message object of used command.
- **custom_stats** (dict) – Dict with custom server stats. (Optional)

3.3 Users

class boticordpy.modules.**Users**(**kwargs)

Class with methods to work with Boticord API Users.

async **getUserBots**(userID: int)

Returns bots of discord user with the given ID.

Parameters **userID** (int) – Discord User's ID

async **getUserComments**(userID: int)

Returns comments of discord user with the given ID.

Parameters **userID** (int) – Discord User's ID

async **getUserInfo**(userID: int)

Returns information about discord user with the given ID.

Parameters **userID** (int) – Discord User's ID

BOTICORDWEBHOOK

```
class boticordpy.BoticordWebhook(bot: Union[discord.ext.commands.bot.Bot,  
                                         discord.ext.commands.bot.AutoShardedBot], boticord_client:  
                                         boticordpy.client.BoticordClient)
```

This class is used as a manager for the Boticord webhook.

:param bot `commands.Bot` | `commands.AutoShardedBot`: The discord.py Bot instance

```
bot_webhook(route: str = '/bot', hook_key: str = '') → boticordpy.webhook.BoticordWebhook
```

This method may be used to configure the route of boticord bot's webhook.

:param route `str`: Bot's webhook route. Must start with /. Defaults - /bot. :param hook_key `str`:
Webhook authorization key.

Returns

Return type `BoticordWebhook`

```
async close() → None
```

Stops the webhook.

```
run(port: int)
```

Runs the webhook.

Parameters `port` – The port to run the webhook on.

REFERENCE

5.1 Event Reference

Example of event creation:

```
from discord.ext import commands

from boticordpy import BoticordWebhook, BoticordClient

bot = commands.Bot(command_prefix="!")
boticord = BoticordClient(bot, "boticord-api-token")

boticord_webhook = BoticordWebhook(bot, boticord).bot_webhook("/bot", "X-Hook-Key")
boticord_webhook.run(5000)

@boticord.event("edit_bot_comment")
async def on_boticord_comment_edit(data):
    print(data)
```

You can name the function whatever you want, but the decorator must always specify an existing event as an argument.

Warning: All the events must be a **coroutine**. If they aren't, then you might get unexpected errors. In order to turn a function into a coroutine they must be `async def` functions.

Here you can find some information about events:

| Boticord Events | Returns Type |
|-----------------------|---------------------|
| new_bot_comment | types.Comment |
| edit_bot_comment | types.EditedComment |
| delete_bot_comment | types.Comment |
| new_bot_bump | types.BotVote |
| new_server_comment | Raw Data |
| edit_server_comment | Raw Data |
| delete_server_comment | Raw Data |

You can find more events in [Boticord Documentation](#).

5.2 Types

class boticordpy.types.**BotVote**(*raw_data*)

Model that represents information about bot's vote.

raw_data

Raw data from the Boticord API.

Type dict

user_id

ID of user, who voted.

Type int

at

Voting date.

Type datetime.datetime

class boticordpy.types.**Comment**(*raw_data*)

Model that represents information about a comment.

raw_data

Raw data from the Boticord API.

Type dict

user_id

ID of comment author.

Type int

comment

Comment.

Type str

at

The comment creation time.

Type datetime.datetime

class boticordpy.types.**CommentData**(*raw_data*)

Model that represents edited comment text data.

old

Old comment text.

Type str or None

new

New comment text.

Type str or None

class boticordpy.types.**EditedComment**(*raw_data*)

Model that represents information about edited comment. It is inherited from [Comment](#)

comment

Comment.

Type [CommentData](#)

EXCEPTIONS

This page describes all the exceptions of boticordpy module.

exception `boticordpy.exceptions.BoticordException`

Base exception class for boticordpy. This could be caught to handle any exceptions thrown from this library.

exception `boticordpy.exceptions.Forbidden(response)`

Exception that's thrown when status code 403 occurs.

exception `boticordpy.exceptions.HTTPException(response)`

Exception that's thrown when an HTTP request operation fails.

response

The response of the failed HTTP request.

message

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

exception `boticordpy.exceptions.NotFound(response)`

Exception that's thrown when status code 404 occurs.

exception `boticordpy.exceptions.ServerError(response)`

Exception that's thrown when status code 500 or 503 occurs.

exception `boticordpy.exceptions.ToManyRequests(response)`

Exception that's thrown when status code 429 occurs.

exception `boticordpy.exceptions.Unauthorized(response)`

Exception that's thrown when status code 401 occurs.

LINKS

- [search](#)
- [PyPi](#)
- [GitHub](#)
- [Boticord](#)

PYTHON MODULE INDEX

b

`boticordpy.exceptions`, [13](#)

`boticordpy.types`, [12](#)

INDEX

A

at (*boticordpy.types.BotVote* attribute), 12
at (*boticordpy.types.Comment* attribute), 12

B

bot_webhook() (*boticordpy.BoticordWebhook* method), 9
BoticordClient (*class in boticordpy*), 5
BoticordException, 13
boticordpy.exceptions
 module, 13
boticordpy.types
 module, 12
BoticordWebhook (*class in boticordpy*), 9
Bots (*boticordpy.BoticordClient* attribute), 5
Bots (*class in boticordpy.modules*), 7
BotVote (*class in boticordpy.types*), 12

C

close() (*boticordpy.BoticordWebhook* method), 9
comment (*boticordpy.types.Comment* attribute), 12
comment (*boticordpy.types.EditedComment* attribute), 12
Comment (*class in boticordpy.types*), 12
CommentData (*class in boticordpy.types*), 12

E

EditedComment (*class in boticordpy.types*), 12
event() (*boticordpy.BoticordClient* method), 5

F

Forbidden, 13

G

getBotComments() (*boticordpy.modules.Bots* method), 7
getBotInfo() (*boticordpy.modules.Bots* method), 7
getServerComments() (*boticordpy.modules.Servers* method), 7
getServerInfo() (*boticordpy.modules.Servers* method), 7
getUserBots() (*boticordpy.modules.Users* method), 8

getUserComments() (*boticordpy.modules.Users* method), 8
getUserInfo() (*boticordpy.modules.Users* method), 8

H

HTTPException, 13

M

message (*boticordpy.exceptions.HTTPException* attribute), 13
module
 boticordpy.exceptions, 13
 boticordpy.types, 12

N

new (*boticordpy.types.CommentData* attribute), 12
NotFound, 13

O

old (*boticordpy.types.CommentData* attribute), 12

P

postServerStats() (*boticordpy.modules.Servers* method), 7
postStats() (*boticordpy.modules.Bots* method), 7

R

raw_data (*boticordpy.types.BotVote* attribute), 12
raw_data (*boticordpy.types.Comment* attribute), 12
response (*boticordpy.exceptions.HTTPException* attribute), 13
run() (*boticordpy.BoticordWebhook* method), 9

S

ServerError, 13
Servers (*boticordpy.BoticordClient* attribute), 5
Servers (*class in boticordpy.modules*), 7
start_loop() (*boticordpy.BoticordClient* method), 5

T

ToManyRequests, 13

U

Unauthorized, [13](#)

user_id (*boticordpy.types.BotVote attribute*), [12](#)

user_id (*boticordpy.types.Comment attribute*), [12](#)

Users (*boticordpy.BoticordClient attribute*), [5](#)

Users (*class in boticordpy.modules*), [8](#)