ib3

Release 0.1.2.dev9+g3e44b48.d20190107

Jan 07, 2019

Contents

IRC bot framework using mixins to provide commonly desired functionality.

CHAPTER 1

IRC Bot Behavior Bundle (IB3)

IRC bot framework using mixins to provide commonly desired functionality.

1.1 Overview

The irc python library's `irc.bot.SingleServerIRCBot` provides a nice base for making a new bot, but there are many common tasks needed by a robust bot that it does not handle out of the box. IB3 collects some commonly desired behaviors for a bot as mixin classes that can be used via multiple inheritance:

```
from ib3 import Bot
from ib3.auth import SASL
from ib3.connection import SSL
from ib3.mixins import DisconnectOnError

class TestBot(SASL, SSL, DisconnectOnError, Bot):
    pass
```

1.2 Installation

- `pip install ib3` (recommended)
- `python setup.py install` (from source distribution)

1.3 License

IB3 is licensed under the GNU GPLv3+ license.

1.4 Credits

Some code and much inspiration taken from Wikimedia irc bots Adminbot, Jouncebot, and Stashbot.

CHAPTER 2

ib3 package

2.1 Module contents

class ib3.**Bot**(**args*, ***kwargs*)

Bases: irc.bot.SingleServerIRCBot

Basic IRC bot.

Simple subclass of `irc.bot.SingleServerIRCBot` that sets up lenient UTF-8 encoding handling for inbound messages. This is a nice base to start from when adding other IB3 mixins.

__init__(**args*, ***kwargs*)

x.__init__(...) initializes x; see help(type(x)) for signature

2.2 Submodules

2.3 ib3.auth module

class ib3.auth.**AbstractAuth**(*server_list*, *nickname*, *realname*, *ident_password*, *channels=[]*, *username=None*, ***kwargs*)

Bases: *ib3.mixins.JoinChannels*

Base class for authentication mixins.

__init__(*server_list*, *nickname*, *realname*, *ident_password*, *channels=[]*, *username=None*, ***kwargs*)

Parameters

- **server_list** – List of servers the bot will use.
- **nickname** – The bot's nickname
- **realname** – The bot's realname

- **ident_password** – The bot's password
- **channels** – List of channels to join after authenticating
- **username** – IRC username (default: nickname)

class ib3.auth.**NickServ**(**args*, ***kwargs*)
Bases: *ib3.auth.AbstractAuth*

Authenticate with NickServ before joining channels.

__init__(**args*, ***kwargs*)

Parameters

- **server_list** – List of servers the bot will use.
- **nickname** – The bot's nickname
- **realname** – The bot's realname
- **ident_password** – The bot's password
- **channels** – List of channels to join after authenticating
- **username** – IRC username (default: nickname)

_handle_privnotice(*conn*, *event*)
Handle NOTICE sent directly to user.

Check for messages from NickServ requesting auth, warning of password failures, and acknowledging successful auth.

_handle_welcome(*conn*, *event*)
Handle WELCOME message.

Starts authentication handshake by sending NickServ an `identify` message.

_identify_to_nickserv()
Send NickServ our username and password.

class ib3.auth.**SASL**(**args*, ***kwargs*)
Bases: *ib3.auth.AbstractAuth*

Authenticate using SASL before joining channels.

__init__(**args*, ***kwargs*)

Parameters

- **server_list** – List of servers the bot will use.
- **nickname** – The bot's nickname
- **realname** – The bot's realname
- **ident_password** – The bot's password
- **channels** – List of channels to join after authenticating
- **username** – IRC username (default: nickname)

_handle_903(*conn*, *event*)
Handle 903 RPL_SASLSUCCESS responses.

_handle_908(*conn*, *event*)
Handle 908 RPL_SASLMECHS responses.

_handle_authenticate(*conn*, *event*)
Handle AUTHENTICATE responses.

_handle_cap(*conn*, *event*)
Handle CAP responses.

_handle_connect(*sock*)
Send CAP REQ :sasl on connect.

_handle_welcome(*conn*, *event*)
Handle WELCOME message.

2.4 ib3.connection module

class ib3.connection.**SSL**(**args*, ***kwargs*)
Bases: object

Use SSL connections.

__init__(**args*, ***kwargs*)
x.__init__(...) initializes x; see help(type(x)) for signature

2.5 ib3.mixins module

class ib3.mixins.**DisconnectOnError**(**args*, ***kwargs*)
Bases: object

Handle ERROR message by logging and disconnecting.

__init__(**args*, ***kwargs*)
x.__init__(...) initializes x; see help(type(x)) for signature

class ib3.mixins.**JoinChannels**
Bases: object

Join channels one at a time to avoid flooding.

join_channels(*channels*)
Join a list of channels, one at a time.

class ib3.mixins.**PingServer**(*max_pings=2*, *ping_interval=300*, **args*, ***kwargs*)
Bases: object

Add checks for connection liveness using PING commands.

__init__(*max_pings=2*, *ping_interval=300*, **args*, ***kwargs*)

Parameters

- **max_pings** – Maximum numer of missed pings to tolerate
- **ping_interval** – Seconds between ping attempts

_handle_pong(*conn*, *event*)
Clear ping count when a pong is received.

_ping_server()
Send a ping or disconnect if too many pings are outstanding.

class ib3.mixins.**RejoinOnBan**(**args*, ***kwargs*)
Bases: object

Handle ERR_BANNEDFROMCHAN by attempting to rejoin channel.

__init__(**args*, ***kwargs*)
x.__init__(...) initializes x; see help(type(x)) for signature

class ib3.mixins.**RejoinOnKick**(**args*, ***kwargs*)
Bases: object

Handle KICK by attempting to rejoin channel.

__init__(**args*, ***kwargs*)
x.__init__(...) initializes x; see help(type(x)) for signature

2.6 ib3.nick module

class ib3.nick.**Ghost**(*server_list*, *nickname*, *realname*, *altnick=None*, **args*, ***kwargs*)
Bases: *ib3.nick.NicknameInUse*

GHOST disconnects an old user session, or somebody attempting to use your nickname without authorization.

This mixin assumes that you are also using a mixin from ib3.auth or another means to authenticate your IRC account.

_recover_nick()
Recover nick by sending GHOST command to NickServ.

class ib3.nick.**NicknameInUse**(*server_list*, *nickname*, *realname*, *altnick=None*, **args*, ***kwargs*)
Bases: object

Handle ERR_NICKNAMEINUSE messages by changing to an alternate nick.

__init__(*server_list*, *nickname*, *realname*, *altnick=None*, **args*, ***kwargs*)

Parameters

- **server_list** – List of servers the bot will use.
- **nickname** – The bot's nickname
- **realname** – The bot's realname
- **altnick** – Alternate nickname if primary nick is taken

_handle_nicknameinuse(*conn*, *event*)
Handle ERR_NICKNAMEINUSE message.

If failed nick matches our desired nick, switch to secondary nick, and schedule an attempt to reclaim our primary nick.

If failed nick matches our secondary nick, die.

_recover_nick()
Do nothing.

Subclasses can override this method to perform some special action to recover the bot's primary nickname.

has_primary_nick()
Do we currently have our primary nick?

Return type bool

class ib3.nick.**Regain**(*server_list*, *nickname*, *realname*, *altnick=None*, **args*, ***kwargs*)

Bases: *ib3.nick.NicknameInUse*

REGAIN disconnects an old user session, or somebody attempting to use your nickname without authorization, then changes your nickname to the given nickname. This may not work, disconnecting you, if the target client reconnects automatically.

This mixin assumes that you are also using a mixin from `ib3.auth` or another means to authenticate your IRC account.

_recover_nick()

Recover nick by sending REGAIN command to NickServ.

Python Module Index

i

Index

Symbols

A

B

D

G

H

I

J

N

P

R

S