

twbot2.rb - Twitter Bot Support Library in Ruby

(C) 2010- H.Hiro(Maraigue)

main@hhiro.net / <http://maraigue.hhiro.net/twbot/>

目次

1. はじめに.....	1
2. bot を動かすための準備.....	2
3. bot の挙動を定義するコードの書き方.....	5
4. 旧バージョン (twbot.rb) からの移行.....	6
5. 簡易リファレンス.....	7
6. 詳細リファレンス.....	8
7. 著作権表示.....	16

1. はじめに

1.1 ライブラリの概要

twbot2.rb は、Ruby にて Twitter の bot (プログラムにより自動的に発言を行うアカウント) 作成を行う際に、その補助を行うフレームワーク・ライブラリです。

特に、定期的に特定の位置からデータを取得し (RSS や Twitter のタイムラインなど)、それに基づいて発言を行う bot を作るのに適したものとなっています。

1.2 必要な環境

Ruby インタプリタの導入されている、定期的にプログラムを実行させるためのコンピュータが必要です。

- 定期的なプログラムの実行を行う機構としては、Unix 系 OS であれば cron、Windows であればタスクスケジューラなどがあります。(これらに関する詳細な説明は行いません)
- Ruby は 1.8 および 1.9 の利用を推奨します。Ruby1.8.7, 1.9.1 でテストしています。
 - Ruby の OAuth ライブラリを別途インストールする必要があります。 Ruby1.8 および 1.9 の場合、別途必要なライブラリはこれのみです。
<http://oauth.rubyforge.org/>
 - Ruby1.7 以前での稼働は推奨しません。(yaml、rexml ライブラリが必要なため)

Table of contents

1. Introduction	1
2. How to start a bot	2
3. Instructions for defining bot's behavior	5
4. How to update from old library (twbot.rb)	6
5. Simple references	7
6. Detailed references	8
7. Copyright	16

1. Introduction

1.1 Overview of the library

twbot2.rb is a framework / library for Ruby programming language, which supports making Twitter bots.

twbot2.rb especially help you much make Twitter bots that get data periodically from certain place (RSS, Twitter timeline, etc.) and create tweets from that data.

1.2 Requirements

A computer with Ruby interpreter, for executing a program repeatedly is required.

- To run programs repeatedly, you can use "cron" system in Unix(-like) OSes, or Task Manager in Windows. (This document does not explain those systems in detail.)
- This library works on Ruby 1.8 or 1.9. It is tested with Ruby 1.8.7 and 1.9.1.
 - OAuth library for Ruby is needed.
<http://oauth.rubyforge.org/>
If you use Ruby 1.8 or 1.9, this is the only library to be installed.
 - Using this library under Ruby 1.7 or older is not recommended, because twbot.rb needs "yaml" and "rexml" library.

2. bot を動かすための準備

2.1 準備

ファイル「twbot2.rb」、ディレクトリ「twbot2」、およびあなたの bot の動作を定義する Ruby スクリプトを、同一のディレクトリに配置します。ここでは、bot の動作を定義するスクリプトとして、同梱されているサンプル「simplebot.rb」を用いた場合について説明します。

2.2 bot アカウントの認証

まず、コマンドラインから

```
ruby simplebot.rb init
```

と入力します。すると以下のような表示が出ます。

```
=====
Here I help you register your bot account to the setting file.
Please prepare a browser to retrieve OAuth tokens.

Input the screen name of your bot account.
=====
User name >
```

ここで、bot として使うアカウントの名前を入力します。もしアカウントを準備していないなら、あなたのアカウントを指定してもよいでしょう。

入力すると続いて、以下のような表示が出ます。

(ただし、「*****」には入力したユーザ名、「#####」にはランダムな文字列が表示されます。)

2. How to start a bot

2.1 Preparation

Locate the following files and directory in a directory: file “twbot2.rb”, directory “twbot2” and a Ruby script to define the bot’s behavior. In the example below, The attached example “simplebot.rb” is used as the script for bot’s behavior.

2.2 Authenticating the bot account

Input the following command from a command line:

```
ruby simplebot.rb init
```

Then you will get the response like this:

Then input the account name of your bot. If you don’t have an account for a bot, you may input your account name.

After input you will get the response like this, where ***** is the account name you have input and ##### is a random string.

```
=====
To retrieve OAuth token of user "*****":
(1) Log in Twitter with a browser for user "*****".
(2) Access the URL below with same browser:
    https://api.twitter.com/oauth/authorize?oauth_token=#####
(3) Check the application name is "twbot2.rb" and
    click "Allow" link in the browser.
(4) Input the shown number (PIN number).
    To cancel, input nothing and press enter key.
=====
PIN number >
```

ここで、認証のためにはブラウザが必要となります。(1)から(4)の手順に従って認証を行って下さい。

Here you have to use a browser. Follow the message from (1) to (4) to authenticate.

2.2 bot アカウントの認証 (続き)

- (1) まずブラウザを用いて、いま入力したユーザで Twitter にログインします。
- (2) そのブラウザで、上記画面で指定された URL にアクセスします。
- (3) 表示されたページにて、アクセスを求めているアプリケーションが「twbot2.rb」であることを確認し、「許可する」のリンクをクリックします。
- (4) その後暗証番号が表示されるので、それを「PIN number」の欄に入力します。

正しい番号を入力した場合、User "*****" is successfully registered. と表示されます。これで認証は完了です。

このとき「config.yml」というファイルが生成されているので、テキストエディタで開いて下さい。以下の様な部分があれば成功です。

```
users/*****:  
  token: #####  
  secret: #####
```

2.3 発言の生成と投稿

コマンドラインに戻り、

```
ruby simplebot.rb load
```

と入力します。すると以下のような表示が出ます。(時刻は実際の実行時刻になります)

```
[Sun May 01 00:00:00 +0000 2010 mode=load]
```

この表示のみが現れた場合は成功です。失敗した場合はエラー情報が同時に表示されます。

成功した場合、設定ファイルに発言すべき内容が書き込まれています。「config.yml」をテキストエディタで開き、以下のような部分があれば成功です。(「20」という数は異なっている場合もあります)

```
data/  
- 20 new replies found!
```

その後改めてコマンドラインに戻り、

```
ruby simplebot.rb post
```

と入力します。すると以下のような表示が出ます。

```
[Updated!] 20 new replies found!  
[Sun May 01 00:01:00 +0000 2010 mode=post] (A tweet has posted)
```

このように表示された場合、投稿は成功しています。bot アカウントのページを開き、投稿があることを確認して下さい。

2.2 Authenticating the bot account (continued)

If you input correct PIN number, you will get a response User "*****" is successfully registered. If so, the account is successfully authenticated and a file "config.yml" is created. You will find a notation like below in "config.yml":

2.3 Generating and posting messages

Go back to command line and input:

```
ruby simplebot.rb load
```

Then you will get the response like this. (The shown time is when you ran the program.)

If succeeded, you will get only the message. If failed, error information will be shown together.

If succeeded, the message to be posted to Twitter is written to the configuration file. You will find a notation like below in "config.yml". (The number "20" may differ.)

Then go back to command line again and input:

```
ruby simplebot.rb post
```

Then you will get the response like this:

If you get the result, you have succeeded in posting a message to Twitter. Open the account's page and check the message is posted.

2.4 bot の継続的な稼働

この bot を継続的に動作させるには、以下のコマンドが定期的に呼ばればよいです。

```
ruby simplebot.rb load post
```

このコマンドが実行されるたび、発言の生成と投稿が一度ずつ行われます。

Windows であれば、タスクスケジューラで実行されるべきプログラムを上記のものにすればよいです。

Unix 系 OS (Mac OS X 含む) であれば、cron で

```
ruby simplebot.rb load post > /dev/null  
2>&1
```

を実行すればよいです。

(詳細な説明は省きます)

2.4 Run the bot continuously

To run the bot continuously, the following command should be called periodically:

```
ruby simplebot.rb load post
```

For each call of the command, one message is created and posted.

If you use Windows machine, add the command to be run to the task scheduler.

If you use Unix(-like) OS, including Mac OS X, then add a command to cron:

```
ruby simplebot.rb load post > /dev/null  
2>&1
```

For detailed instruction, see the document of the each function.

3. bot の挙動を定義するコードの書き方

コードは以下の手順で書かれます。

1. `TwBot` クラスのサブクラス (`TwBot` を継承したクラス) を定義する。
2. そのサブクラスで `load_data` メソッドを再定義する。
 - `load_data` メソッドは、その bot が発言する内容の文字列を配列にして返すメソッドです。それぞれの文字列が 1 つの発言になり、また先頭にある要素ほど先に投稿されます (キューの形になります)。
 - 配列の要素は、文字列ではなく配列やハッシュも認められます。詳細は `TwBot#load_data` のリファレンスをご覧ください。
3. そのサブクラスのインスタンスを生成する。このとき、`TwBot.new` の第 1 引数の値に応じて行われるべき処理が決まる。例えば
 - "load" を与えると、`load_data` メソッドを呼び出してその bot が発言する内容をリストに追加し、投稿待ちの状態にします。
 - "post" を与えると、リストの先頭の発言を取り出し、Twitter に投稿します。

詳細はリファレンスをご覧ください。またコードの例は、同梱の "simplebot.rb" をご覧ください。

3. Instructions for defining bot's behavior

A code for your bot's behavior is to be written by the procedure below:

1. Define a subclass of the class `TwBot` (a class inheriting `TwBot`).
2. Re-define the `load_data` method of the subclass.
 - `load_data` should return an array of strings to be tweeted. Each string will be a tweet. The strings are tweeted with order of the array. (The messages are queued.)
 - The array to be returned may contain arrays or hash-tables. See the reference of `TwBot#load_data` for details.
3. Create an instance of the subclass. The first argument of `TwBot.new` decides what to do. For example,
 - If the first argument is "load", `load_data` method is called and new messages to be tweeted are generated and queued.
 - If "post", the top message of the queue is fetched and tweeted in Twitter.

See the reference for details. See a sample of a code for the attached file "simplebot.rb"

4. 旧バージョン (twbot.rb) からの移行

旧バージョン (twbot.rb) 向けに書かれたコードを twbot2.rb に移行するには、以下の手順を踏んで下さい。

1. 「2.2 bot アカウントの認証」の節で書かれた手順を参考に、bot アカウントの認証を行って下さい。より厳密には、「3. bot の挙動を定義するコードの書き方」で述べた「クラスのインスタンスを生成する」にあたり、「TwBot.new」の第1引数の値の値を「init」として呼び出して下さい。この場合においては、「2.2 bot アカウントの認証」の節で書かれた手順と異なり、ユーザ名の入力は不要です (自動検出されます)。
2. 互換性のなくなったコードを置き換えます。下の対応表をご覧ください。
3. Twitter への HTTP アクセスに OAuth ではなく BASIC 認証を使っているメソッドを置き換えます。具体的には、`auth_http(username)` を使うことで OAuth 認証付きの HTTP アクセスを行うことが出来ます。詳細はリファレンスをご覧ください。

4. How to update from old library (twbot.rb)

To update the old versions (twbot.rb) to twbot2.rb, please take the procedure below:

1. Get the bot account to be authenticated, by way like the section 2.2 "Authenticating the bot account". Strictly, give "init" as the first argument of `TwBot.new`, described in the section 3 "Instructions for defining bot's behavior".
In this case you don't have to input the account name of the bot (automatically detected), unlike the way shown in the section 2.2.
2. Replace codes deprecated in twbot2.rb. See the table below for the removed functions in twbot2.rb.
3. Replace codes with BASIC authentication to OAuth for HTTP accesses to Twitter.
You can conduct an OAuth-authenticated HTTP access by `auth_http(username)`. See the reference for details.

Removed functions	New functions
<code>TwBot.retrieve_followers(login, pass, retry_count)</code>	<code>TwBot.followers_of(auth_http(login), retry_count)</code>
<code>TwBot.retrieve_friends(login, pass, retry_count)</code>	<code>TwBot.friends_of (auth_http(login), retry_count)</code>
<code>TwBot.make_following(login, pass, user)</code>	<code>follow(user, auth_http(login))</code>
<code>TwBot.make_unfollowing(login, pass, user)</code>	<code>unfollow(user, auth_http(login))</code>
<code>TwBot.check_following(login, pass, target)</code>	<code>following_status(target, auth_http(login))</code>

5. 簡易リファレンス

bot の定義部 (`TwBot#load_data` 内) でよく用いられるコードの一覧です。

- bot アカウントのタイムラインを取得する

```
xml = auth_http.get("/1/statuses/home_timeline.xml").body
```

※末尾の「.xml」は、「.json」でもよい。この場合発言は JSON 形式で取得される。以下同様。

※返ってくる XML および JSON のフォーマットは、以下の URL を参照。

[http://apiwiki.twitter.com/Twitter-REST-API-Method%3A-statuses-home timeline](http://apiwiki.twitter.com/Twitter-REST-API-Method%3A-statuses-home+timeline)

- bot アカウントへの返信を取得する

```
xml = auth_http.get("/1/statuses/mentions.xml").body
```

- 特定のユーザについて、最新の発言を取得する

```
xml = auth_http.get("/1/statuses/user_timeline/h_hiro_.xml").body
```

※@h_hiro_の発言を取得する場合。

- bot アカウントを (bot アカウントが) フォローしているユーザー一覧を返す

```
followers = get_followers[:result] # list of users the bot account is followed by  
friends = get_friends[:result] # list of users the bot account follows
```

- ユーザをフォローする/フォロー解除する

```
follow("h_hiro_") # to follow the user "h_hiro_"  
unfollow("h_hiro_") # to unfollow the user "h_hiro_"
```

`auth_http` を用いて呼び出している機能は、すべて Twitter が提供している API です。これ以外にも多数の機能がありますので、詳細は以下の URL をご覧下さい。

<http://apiwiki.twitter.com/Twitter-API-Documentation>

5. Simple references

Codes often used for defining bots' behaviors (i.e. in `TwBot#load_data`) are shown here.

- Retrieving timeline of the bot account

* ".xml" in the bottom can be replaced by ".json". In this case you will get the result by JSON format.

* See the site below for the format of the returned value (XML or JSON).

- Retrieving mentions of the bot account

- Retrieving tweets of a specified user

* An example of retrieving @h_hiro_'s tweets.

- Listing up users (following/followed by) the bot account

- Following/Unfollowing a user

All functions called via `auth_http` are all APIs served by Twitter. See the URL below for many other functions:

6. 詳細リファレンス

6.1 TwBot クラスのインスタンス変数

`@config #=> <Hash>`

設定ファイルに書き込まれた内容は、この変数に保持されます。

この Hash の一部のキーは、当ライブラリで用いるため決まった名前が付けられています。詳細は「6.1.1 @config 変数において用いられる定義済みキー」をご覧ください。

@config に情報を追加したい場合は、Hash のキーを増やすことで対応して下さい。またその際、半角スラッシュ (「/」) を含む文字列はキーにしないで下さい (将来的に定義済みキーが増えた際、その名前と競合する可能性があるため)。

load_data メソッド中で @config の値を書き換えると、メソッドの処理が完了した際、その変更が設定ファイルに反映されます。ただし load_data メソッドを例外で抜けた場合は、変更は反映されません。

`@list #=> <String>`

発言一覧を格納したリストが、@config のどのキーに割り当てられているかを示す文字列です。

この値は TwBot.new の引数「:list」によって変化し、「data/*****」という形を取ります。ここで「*****」は引数「:list」の値です (よって、デフォルトでは @list == "data/" となります)。

`@logmsg #=> <String>`

ログファイルに書き込まれる文字列を指定します。この値が load_data メソッド中で変更され、かつ load_data メソッドを例外で抜けてはいない場合、この文字列がログファイルに書き込まれます。

6. Detailed references

6.1 Instance variables of class TwBot

`@config #=> <Hash>`

This variable stores the content in the setting file.

Some of the hash keys in it are reserved for the library. See the section 6.1.1 "Reserved keys of the variable @config" for details.

You can add information stored in @config by adding its hash keys. If you do so, use a key name without slashes ("/") so as not to make conflicts to additions reserved keys in future versions.

Changes of @config in load_data method are reflected to the setting file, unless you have left the method by an exception.

`@list #=> <String>`

This variable stores which key of @config should be treated as the message list.

The value is changed by the argument ":list" of the method TwBot.new, and has the format of "data/*****", where "*****" is the value of argument ":list". (By default, @list == "data/.")

`@logmsg #=> <String>`

You can specify what text should be written to the log file. If you change the value in load_data method and the method ended by other than exceptions, then the value of the variable is written to the log file.

6.1.1 @config 変数において用いられる定義済みキー

```
@config["login/"] #=> <String>
```

作成した bot の発言は、このユーザから行われます。なお bot のアカウント名を変更した場合は、設定ファイルの当項目を修正したのち、「2.2 bot アカウントの認証」の節で書かれた手順を参考に、bot アカウントの認証を行って下さい。

```
@config['duplicated/'] #=> <String>
```

重複投稿制限（直近のものと同一内容のツイートをしようとするとう失敗する）が発生した場合の対応方法を指定します。TwBot#update_from_list の引数「:duplicated」のデフォルト値として利用されます。詳細は TwBot#update_from_list をご覧下さい。

```
@config["users/*****"] #=> <Hash>
```

ユーザ「*****」に対応する OAuth トークン (token および secret) を示します。bot アカウントの認証を行うと自動的に設定されます。

```
@config["data/*****"] #=> <Array>
```

投稿待ちの発言一覧を格納するリストです。「*****」は TwBot.new の引数「:list」の値に相当します。通常は@config[@list]の形で利用します。この配列の要素は、String、Array、Hash のいずれかです。詳細は TwBot#load_data の戻り値についての説明をご覧下さい。

6.1.1 Reserved keys of the variable @config

```
@config["login/"] #=> <String>
```

This represents which account posts the tweets. If you change this value, authenticate new user with the way shown in the section 2.2 "Authenticating the bot account".

```
@config['duplicated/'] #=> <String>
```

This represents how twbot2.rb behaves when failed in tweeting because of a duplication (Posting a tweet fails if you post same tweet as recent one).

This is the default value of the argument "duplicated" in TwBot#update_from_list. See the reference of TwBot#update_from_list for details.

```
@config["users/*****"] #=> <Hash>
```

This is the OAuth token and secret for the user "*****". If you authenticate a user, the value is automatically added.

```
@config["data/*****"] #=> <Array>
```

This is the list for unsent tweets. "*****" is the value of argument "list" in the method TwBot.new. The form @config[@list] is often used.

Each element of the array should be a String, Array or Hash. See the note of TwBot#load_data about the return value.

6.2 TwBot クラスのインスタンスメソッド

```
TwBot#load data() #=> <Array>
```

このメソッドを、`TwBot` クラスを継承した子クラスでオーバーライドし、その中で `bot` の発言内容を定義して下さい。そのクラスのインスタンスを生成することで、`bot` を動かします。

その際、発言すべき内容の配列を返り値として返して下さい。配列の要素で先頭にあるものから順に発言されます。また、配列の各要素はそれぞれ以下のいずれかにして下さい。

- 配列の要素が `String` の場合
その文字列をそのまま発言します。
- 配列の要素が `Array` の場合
配列の第 1 要素 (`array[0]`) を発言する文字列、第 2 要素 (`array[1]`) を返信先のツイートの ID とみなします。この場合、「in reply to ****」のリンクが付与されます。
- 配列の要素が `Hash` の場合
その値を HTTP リクエストのパラメータとして直接渡します。キーは `Symbol` にして下さい。例えば返り値の配列の要素にハッシュ

```
{:status => "test",  
:in_reply_to_status_id => 100}
```

を指定することと、配列 `["test", 100]` を指定することは同じです。

```
TwBot#auth http(  
  :user => @config["login/"],  
  :reload => false,  
  :browser => false)  
# => <OAuth::AccessToken>  
or  
TwBot#auth_http(  
  user = @config["login/"])  
# => <OAuth::AccessToken>
```

「`:user`」で指定されたユーザに対して、そのユーザに対応する `OAuth` トークン (アクセストークン) を返します。そのユーザを認証する処理を行っていない場合、トークンの取得を試みます。

「`:reload`」が `true` である場合、取得済みのトークンを破棄して、改めてトークンを取得し直します。

「`:browser`」が `true` である場合、トークンを取得するのにブラウザを利用します。なお `version0.20` 現在、`twbot2.rb` は `xAuth` (ブラウザを使わず、ユーザ名とパスワードのみでトークンを取得する) が利用出来ないため、`false` にすると新規トークンの取得は行えません。返り値は `OAuth::AccessToken` のインスタンスであるため、例えば `auth_http().get("/1/statuses/home_timeline.json")` のようにして認証付きのアクセスを行うことが出来ます。詳細は <http://oauth.rubyforge.org/rdoc/> をご覧下さい。

6.2 Instance methods of class TwBot

```
TwBot#load data() #=> <Array>
```

You should override the method in the subclass of `TwBot`, and define what the bot tweets. The bot works by creating an instance of the subclass.

Please return an array from the method. The tweets are posted in order of the array. Every element of the array should be one of the following formats:

- If the element is a `String`:
The string itself will be tweeted.
- If the element is an `Array`:
The first element (`array[0]`) will be tweeted. The second element (`array[1]`) is treated as the target tweet of replying. In this case you can attach "in reply to" link to the tweet.
- If the element is an `Hash`:
The hash is treated as a set of parameters of HTTP access. Keys of the hash should be a `Symbol`. For example, specifying a hash

```
{:status => "test",  
:in_reply_to_status_id => 100}
```

 is equivalent to specifying an array `["test", 100]` as a tweet.

```
TwBot#auth http(  
  :user => @config["login/"],  
  :reload => false,  
  :browser => false)  
# => <OAuth::AccessToken>  
or  
TwBot#auth_http(  
  user = @config["login/"])  
# => <OAuth::AccessToken>
```

This returns an `OAuth` access token of the specified user "`:user`". If the user is not authenticated yet, `twbot2.rb` will try to get the token.

If "`:reload`" is true, the token is newly loaded even if a token for the user is stored.

If "`:browser`" is true, a browser is used for retrieving a token. Note that `:browser => false` doesn't enable us to retrieve a token, since `twbot2.rb` don't support `xAuth` (a scheme of retrieving a token only with the user name and the password) as of version 0.20.

As the return value is an instance of `OAuth::AccessToken`, you can make an HTTP access by like `auth_http().get("/1/statuses/home_timeline.json")`. (See the document <http://oauth.rubyforge.org/rdoc/>.)

6.2 TwBot クラスのインスタンスメソッド (続き)

```
TwBot#user_registered?(user)
#=> <boolean>
```

`user` がすでに認証されているか、すなわち `@config` に `user` の OAuth トークンがすでに登録されているかを `true/false` で返します。

```
TwBot#update_from_list(
  :user => @config["login/"],
  :list => @list,
  :duplicated => "ignore")
```

or

```
TwBot#update_from_list(
  user = @config["login/"])
```

「`:user`」で指定されたアカウントを利用して、指定された「`:list`」のリストから発言を投稿します。通常は `TwBot.new("post", ...)` で投稿を行うため使う必要はありませんが、別途投稿を行いたい場合にはこちらを利用して下さい。

「`:duplicated`」を指定すると、直近のものと同じツイートを投稿しようとして失敗した場合に、どのように対処するかを指定出来ます。以下のいずれかを指定して下さい。

- "seek"
重複投稿制限に引っかかった場合、引っかからない発言が見つかるまでリストを辿ります。重複投稿制限に引っ掛かった発言は、後にこのメソッドを呼び出した際にもう一度投稿されるよう、リストの末尾に戻されます。
- "discard"
"seek"に似ていますが、重複投稿制限に引っ掛かった発言はすべて破棄されます。
- "cancel"
重複投稿制限に引っかかった場合、その時点で発言を行うことをキャンセルします。重複していた発言はリストからは削除されないため、別途対処しない限り、一度重複投稿制限に引っ掛かると発言が行われない状態が続きます。
- "ignore"
"cancel"に似ていますが、重複していた発言はリストから削除されます。このため、"cancel"の場合と異なり、生成される発言が直近の発言と重複したものでなければ、発言が行われないままになることはありません。`:duplicated`に何も指定がない場合のデフォルトです。

6.2 Instance methods of class TwBot (continued)

```
TwBot#user_registered?(user)
#=> <boolean>
```

This returns `true/false` whether the `user` is authenticated, that is, `user`'s OAuth token is registered in `@config`.

```
TwBot#update_from_list(
  :user => @config["login/"],
  :list => @list,
  :duplicated => "ignore")
```

or

```
TwBot#update_from_list(
  user = @config["login/"])
```

This lets `:user` post a tweet from a `:list`, where `:user` is the account name and `:list` is the name of the list. In many cases you don't have to call this method because `TwBot.new("post", ...)` posts a tweet. Call this method if you want to post a tweet other than `TwBot.new`.

`:duplicated` represents the behavior when failing in posting a tweet because of duplicated tweets. Specify one of the followings:

- "seek"
When duplication rejects a tweet, the message list is sought until rejection is avoided. Rejected tweets are stored back to the bottom of the original list and will be posted later.
- "discard"
This is same as "seek", other than the rejected tweets are discarded (not stored in the original list).
- "cancel"
When duplication rejects a tweet, no message is tweeted. Rejected tweet remains the list. If you don't change or delete the rejected tweet, the bot cannot post any tweet because duplication causes every time.
- "ignore"
This is same as "cancel", other than the rejected tweets are discarded (not stored in the original list). Different from "cancel", if you add tweets without duplication, the bot will not stop posting. This is the default behavior if nothing specified to `:duplicated`.

6.2 TwBot クラスのインスタンスメソッド (続き)

```
TwBot#get_followers()  
#=> {:gained_result => <Array>,  
      :gained_id    => <Array>,  
      :result       => <Array>,  
      :id           => <Array>,  
      :error        => <Array>}
```

デフォルトのユーザ (@config["login/"] に登録されたユーザ) について、そのユーザをフォローしているユーザ一覧をハッシュの形で返します。

- 「:result」 および 「:gained_result」 は、ユーザ名一覧を示します。ただしユーザ一覧の取得が取得の途中で失敗した場合、「:result」は nil になり、「:gained_result」は失敗するまでに取得したユーザの一覧となります。
- 「:id」 および 「:gained_id」 は、ユーザの ID (各ユーザに割り当てられた番号) 一覧を示します。「:id」と「:gained_id」との関係は「:result」の場合と同様です。
- 「:error」には、取得途中に発生した例外の一覧が格納されます。

```
TwBot#get_friends() #=> (省略)
```

デフォルトのユーザについて、そのユーザがフォローしているユーザ一覧を返します。戻り値の形式は TwBot#get_followers と同じです。

```
TwBot#follow(  
  target user, auth = auth http())
```

ユーザの認証情報 auth (省略するとデフォルトのユーザ、以下同様) を用いて、そのユーザに target_user をフォローさせます。

```
TwBot#unfollow(  
  target user, auth = auth http())
```

ユーザの認証情報 auth を用いて、そのユーザの target_user のフォローを解除させます。

```
following status(  
  target user, auth = auth http())  
#=> {:following => <boolean>,  
      :followed => <boolean>}
```

ユーザの認証情報 auth を用いて、そのユーザが target_user をフォローしているか/されているかを調べます。

戻り値は Hash で、「:following」には auth が target_user をフォローしているかが、また「:followed」には auth が target_user にフォローされているかが true/false で格納されています。

6.2 Instance methods of class TwBot (continued)

```
TwBot#get_followers()  
#=> {:gained_result => <Array>,  
      :gained_id    => <Array>,  
      :result       => <Array>,  
      :id           => <Array>,  
      :error        => <Array>}
```

This retrieves the users following the default user (the user specified as @config["login/"]). The return value is a hash and each part of the hash represents:

- Both ":result" and ":gained_result" represent the user names. In case retrieving fails halfway, ":result" is nil, while ":gained_result" is the retrieved users until failure.
- Both ":id" and ":gained_id" represent the user's IDs (ID numbers assigned for each user). The relation of them are same as that of ":result".
- ":error" is the list of exceptions raised in retrieving.

```
TwBot#get_friends() #=> (snipped)
```

This retrieves the users the default user follows. The format of the return value is same as that of TwBot#get_followers().

```
TwBot#follow(  
  target user, auth = auth http())
```

This lets the user represented by the authentication information auth (default user if not specified, same in following methods) follow target_user.

```
TwBot#unfollow(  
  target user, auth = auth http())
```

This lets the user represented by the authentication information auth unfollow target_user.

```
following status(  
  target user, auth = auth http())  
#=> {:following => <boolean>,  
      :followed => <boolean>}
```

This retrieves the user represented by the authentication information auth whether he/she follows or followed by target_user.

The return value is a hash where ":following" represents by true/false whether auth follows target_user, and ":followed" the other way around.

6.3 TwBot クラスのクラスメソッド

```
TwBot.new(  
  mode,  
  config_file,  
  # 以下、必須ではない引数  
  :log_file => nil,  
  :list => "",  
  :keep_config => false,  
  :test => false)
```

あるいは

```
TwBot.new(  
  mode,  
  config_file,  
  # 以下、必須ではない引数  
  log_file = nil,  
  list = "",  
  keep_config = false,  
  test = false)
```

ファイル名 `config_file` で指定された設定ファイルを用いて、発言の追加や投稿、その他設定を行います。

- `mode` は以下のいずれかを文字列で指定します。
 - "load": `TwBot#load_data` を呼び出し、リストに発言を追加します。
 - "post": リストに格納された発言を Twitter に投稿します。
"post=5"や"post=5,2"のように、数値を1つないし2つ指定した文字列を与えることも可能です。その場合、1つ目の値は「リストから最大いくつの発言を Twitter に投稿するか」（デフォルトは1）を示し、2つ目の値は「投稿が失敗した際に何回まで再投稿するか」（デフォルトは0）を示します。
 - "add": 設定ファイルに、ユーザ名とその OAuth アクセストークンを追加します。ブラウザが必要になります。
"add=h_hiro_"のように、ユーザ名を指定することも可能です。指定しなかった場合、`twbot2.rb` が入力するよう求めます。
 - "init": `twbot2.rb` で bot を利用し始める（新規に bot を制作する、あるいは旧ライブラリから移行する）際の、設定ファイルの準備を行います。詳細は「2. bot を動かすための準備」の節をご覧ください。ただし、設定ファイルに `@config["login/"]` に相当する値がすでに記述されている場合は、ユーザ名はそれが利用されるため、ユーザ名を別途入力する必要はありません（旧バージョンの仕様を考慮したため）。
また、"init"指定の場合に限り、`config_file` が存在しなくても新規に生成されます。
- `config_file` は設定ファイルのファイル名です。YAML 形式で書かれています。

6.3 Class methods of class TwBot

```
TwBot.new(  
  mode,  
  config_file,  
  # Following arguments are optional  
  :log_file => nil,  
  :list => "",  
  :keep_config => false,  
  :test => false)
```

or

```
TwBot.new(  
  mode,  
  config_file,  
  # Following arguments are optional  
  log_file = nil,  
  list = "",  
  keep_config = false,  
  test = false)
```

This adds or posts tweets, or set up, with the configuration file `config_file` (specified by file name).

- `mode` should be one of the following strings:
 - "load": `TwBot#load_data` is called and tweets are added to a list.
 - "post": A tweet of specified list is posted to Twitter.
You can instead specify a string with one or two numbers, like "post=5" or "post=5,2". If so, the first number represents at most how many tweets are posted to Twitter (1 by default), and the second number how many times `twbot2.rb` retries posting when failed (0 by default).
 - "add": An OAuth token of a user is added to the configuration file. To do this, a browser is needed.
You can instead specify a string with user name like "add=h_hiro_". If not specified, `twbot2.rb` requests you to input.
 - "init": The config file is prepared for first use of `twbot2.rb` for the bot (to create a new bot, or change the library from old versions to `twbot2.rb`). See the section 2 "How to start a bot" for details. If the value of `@config["login/"]` is already specified in the configuration file, you don't need to input user name, for simplicity of changing the library.
Different from other modes, in this case `config_file` is newly created if not exist.
- `config_file` is the name of configuration file. The file is written by YAML format.

6.3 TwBot クラスのクラスメソッド (続き)

- `log_file` はログを出力する先のファイルです。`nil` を指定すると、ファイルには出力されません。
- `list` は、設定ファイルに保存されている発言リストのうち、どれを利用するかを指定します。インスタンス変数 `@list` にはこの値が反映されます。
- `keep_config` が真である場合、処理終了後に設定ファイルを更新せず、元の状態のままにします。(この場合、`TwBot#load_data` で追加された発言も設定ファイルに追加されません。)
- `test` が真である場合、`mode="post"` を指定したときに、実際に Twitter への投稿を行いません。

```
TwBot.remove_reply(str) #=> <String>
```

`str` に含まれる、ユーザへの返信として扱われる文字列 ("@USERNAME" の形をした文字列) を、 "@ USERNAME" の形 (スペースを挟む) に変換します。これは、ユーザへ過剰に返信を行うことを避けるのに用います。

このメソッドにはブロックを与えることもでき、その場合はブロックに「@」を除いたユーザ名が与えられ、それが真を返した場合のみ変換を行います。

```
# 例
puts TwBot.remove_reply(
  "@user1 @user2 Hello!")
# この場合は "@ user1 @ user2 Hello!" が
# 出力されます。

puts TwBot.remove_reply(
  "@user1 @user2 Hello!"){ |u| u =~ /1/ }
# この場合は "@ user1 @user2 Hello!" が
# 出力されます。"user1"がブロック内の
# 条件を満たしているのに対し、"user2"は
# 満たしていないからです。
```

```
TwBot.followers_of(auth, retry count = 3)
# => (省略)
```

OAuth アクセストークン `auth` に対して、そのユーザをフォローしているユーザー一覧を取得します。アクセスが合計 `retry_count` 回失敗した場合、そこで取得を打ち切ります。

返り値については `TwBot#get_followers` をご覧下さい。

```
TwBot.friends_of(auth, retry count = 3)
# => (省略)
```

`TwBot.followers_of` と同様ですが、代わりにそのユーザがフォローしているユーザー一覧を取得します。

6.3 Class methods of class TwBot (continued)

- `log_file` is the file name where log is output. If this is `nil`, logs aren't kept in a file.
- `list` is the list name that tweets are stored in or posted from. The instance variable `@list` this value.
- If `keep_config` is true, the configuration file is not updated after specified procedure is finished. (In this case, tweets added by `TwBot#load_data` are neither stored in the configuration file.
- If `test` is true, tweets are not actually posted to Twitter when `mode="post"`.

```
TwBot.remove_reply(str) #=> <String>
```

This converts substrings contained in `str` that are treated as users' replies (format of "@USERNAME") into the format of "@ USERNAME" (a space added). This method is used to avoid excessive replies.

A block can be given to the method. In this case, the user name (other than "@") is given to the block as an argument, and the reply is converted if the block returned true.

```
# Example:

puts TwBot.remove_reply(
  "@user1 @user2 Hello!")
# This puts "@ user1 @ user2 Hello!".

puts TwBot.remove_reply(
  "@user1 @user2 Hello!"){ |u| u =~ /1/ }
# This puts "@ user1 @user2 Hello!",
# because "user1" meets the condition
# of the block, while "user2" not.
```

```
TwBot.followers_of(auth, retry count = 3)
# => (omitted)
```

For an OAuth access token `auth`, this returns the list of users who follows `auth`. If errors occurred by `retry_count` times while retrieving the user list, retrieving is stopped.

See `TwBot#get_followers` for the format of the return value.

```
TwBot.friends_of(auth, retry count = 3)
# => (omitted)
```

This is similar to `TwBot.followers_of`, except for the return value is the list of users who `auth` follows.

6.4 TwBot クラスの定数・内部クラス

```
TwBot::Consumer #=> <OAuth::Consumer>
```

twbot2.rb の OAuth コンシューマトークンです。このライブラリが、アクセストークン（ユーザに対応するトークン）取得あるいは利用する際は、上記トークンと組み合わせられます。

```
class TwBot::IncompleteConfigError
```

設定ファイルに記述されている内容が不足している場合に発生する例外です。version 0.20 現在、この例外は「xAAuth を利用しようとして、パスワードが設定ファイルに指定されていない場合」のみ発生します。

RuntimeError を継承しています。

```
class TwBot::MessageFormatError
```

load_data の返り値が、所定の形式でなかったときに発生します。詳細は TwBot#load_data の項目をご覧ください。

またこの例外は、TwBot.new で第 1 引数を "load"（発言の生成）ではなく "post"（発言の投稿）にした場合も発生することがあります。設定ファイルを手動で変更して、所定のフォーマットを満たさなくなった場合には、"post" の段階でも発生し得ます。

RuntimeError を継承しています。

6.4 Constants and internal classes of class TwBot

```
TwBot::Consumer #=> <OAuth::Consumer>
```

This is the OAuth consumer token for twbot2.rb. Access tokens (tokens associated with users) twbot2.rb retrieves from or accesses to Twitter are combined with the token.

```
class TwBot::IncompleteConfigError
```

This exception is raised when the setting file doesn't have enough information. As of version 0.20, the exception is raised only when a user's password is not found in the configuration file and xAuth can't be used.

This exception is inherited from RuntimeError.

```
class TwBot::MessageFormatError
```

The exception is raised when the format of load_data's return value is not acceptable. See the reference of TwBot#load_data for the details of the format.

The exception can be raised when "post" is specified as the first argument of TwBot.new, not only "load"; in case the configuration file is changed manually and the format gets invalid.

This exception is inherited from RuntimeError.

7. 著作権表示

7.1 ライセンス

twbot2.rb、ならびに付属する devnull.rb は、いずれも（新）BSD ライセンスのもとで配布します。

簡単に言えば、このソフトは無保証であり、また再配布（改変した上での再配布含む）の場合においては、元の著作権表示などの部分が記されている必要があります。ライセンスの本文は twbot2.rb の末尾に記載してあります。

※BSD ライセンスの本文は英語です。日本語の参考訳は以下のページをご覧ください。

[new BSD license - Open Source Group Japan Wiki](#)

7.2 お問い合わせ

メール: main@hhiro.net

Twitter: http://twitter.com/h_hiro

7. Copyright

7.1 License

"twbot2.rb" and the attached file "devnull.rb" are both distributed under the (new) BSD License.

Simply,

Simply, this script is distributed under no warranty, and re-distributed version has to contain original copyright information. The license article itself is written in the bottom of the file "twbot2.rb".

7.2 Contact the author

mail: main@hhiro.net

Twitter: http://twitter.com/h_hiro

(Mainly Japanese; Feel free to talk in English!)