

slackbotの導入と基本的な使い方

更新日時 2017/08/19

前提条件

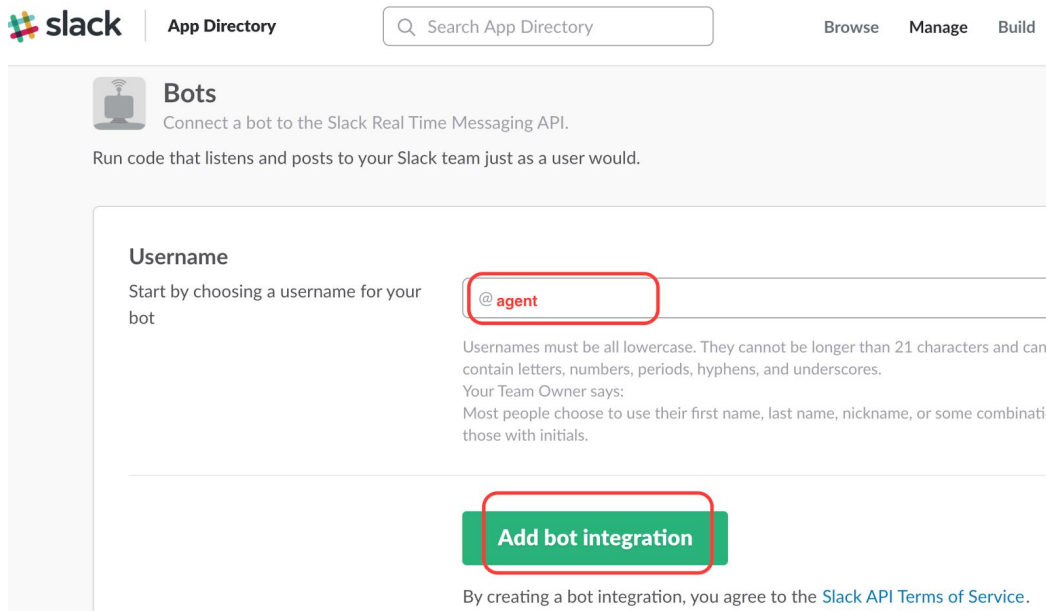
- ・ python3 を利用
- ・ Ubuntu16.04 を利用
- ・ pip 導入済み
- ・ 既にSlackは開設済で、権限のあるユーザでログインしている。

1. Slack Botアカウントの作成

下記リンクを開くと、Botアカウントの作成画面が開きます。

<https://my.slack.com/services/new/bot>

以下の例では agent と入力→「Add bot integration」を押下



slack | App Directory

Search App Directory

Browse Manage Build

Bots
Connect a bot to the Slack Real Time Messaging API.
Run code that listens and posts to your Slack team just as a user would.

Username
Start by choosing a username for your bot

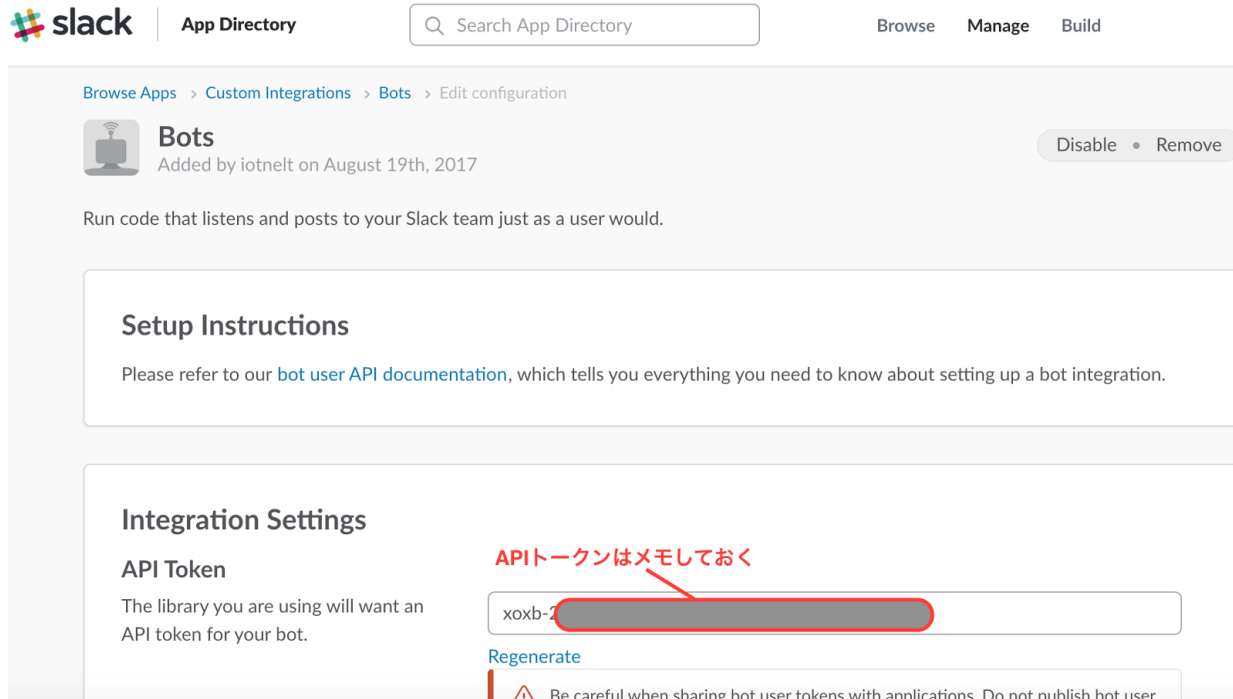
@agent

Username must be all lowercase. They cannot be longer than 21 characters and can contain letters, numbers, periods, hyphens, and underscores.
Your Team Owner says:
Most people choose to use their first name, last name, nickname, or some combination of those with initials.

Add bot integration

By creating a bot integration, you agree to the [Slack API Terms of Service](#).

以下の画面に遷移



slack | App Directory

Search App Directory

Browse Manage Build

[Browse Apps](#) > [Custom Integrations](#) > [Bots](#) > Edit configuration

Bots
Added by iotnelt on August 19th, 2017

Disable • Remove

Run code that listens and posts to your Slack team just as a user would.

Setup Instructions
Please refer to our [bot user API documentation](#), which tells you everything you need to know about setting up a bot integration.

Integration Settings

API Token
The library you are using will want an API token for your bot.

xoxb-2

Regenerate

⚠ Be careful when sharing bot user tokens with applications. Do not publish bot user

[Regenerate](#)



Be careful when sharing bot user tokens with applications. Do not publish bot user tokens in public code repositories. [Review token safety tips](#).

Customize Name

Choose the username for this bot.

@agent

Username must be all lowercase. They cannot be longer than 21 characters and can only contain letters, numbers, periods, hyphens, and underscores.

Your Team Owner says:

Most people choose to use their first name, last name, nickname, or some combination of those with initials.

Customize Icon

Change the icon used for this bot.



Upload an image

or

Choose an emoji

Preview Message

Preview Message

Here's what messages from this integration will look like in Slack.



agent

APP 2:12 AM

This is what messages from this service will look like in Slack.

First & Last Name

You can choose a real name for your bot (optional).

What this bot does

Let others on your team know what this bot is used for (optional).

Channels

This bot is currently in the following channels.

neltbottest02 is in no channels.

Save Integration

API Tokenをメモっておく

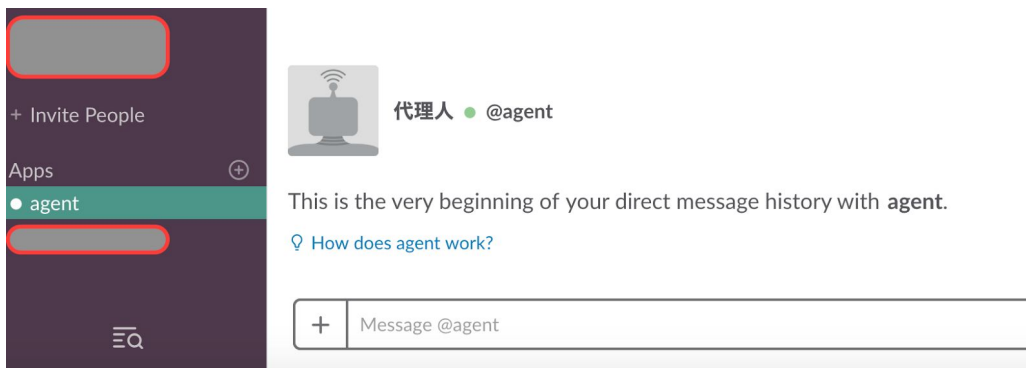
xoxb-228018509936-3f6KtwXL13Tlx3aidbFlyQ0y

ボット名

@agent

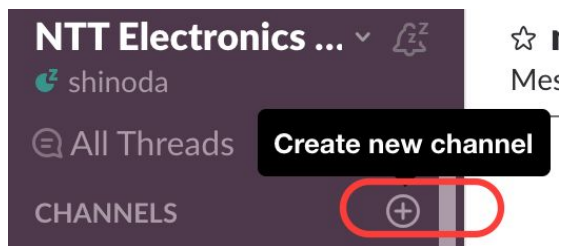
「Customize Icon」や「Customize Name」で作成したBotの名前やアイコン画像を変更できます。変更内容を「Save Integration」で保存します。

slack 画面でボットが追加されたことを確認。



2. Bot登録

テスト用にプライベートチャンネルを作成して作成したBotを登録します。
CHANNELS の右の「+」のボタンを押下。



Create a private channel

Channels are where your team communicates. They're best when organized around a topic — #leads, for example.

☒ Private This channel can only be joined or viewed by invite.

Name

Names must be lowercase, without spaces or periods, and shorter than 22 characters.

Purpose (optional)

What's this channel about?

Send invites to: (optional) 先ほど作成したボット名を選択してメンバに追加する

3. slackbotのインストール

slackbotライブラリをインストール

```
# pip install slackbot
```

```
# pip list|grep slack
```

DEPRECATION: The default format will switch to columns in the future. You can use --format=(legacy|columns) (or define a format=(legacy|columns) in your pip.conf under the [list] section) to disable this warning.

slackbot (0.5.1)

slack (0.9.50)

4. slackbotを動かすための用意

以下のようなディレクトリ構造にします。

```
/var/slackbot
├─ run.py          # ボットを起動する
├─ slackbot_settings.py # botに関する設定を書くファイル
├─ plugins         # botの機能はこのディレクトリに追加する
├─ __init__.py     # モジュールを示すためのファイル。空で良い
└─ my_mention.py   # 機能を各ファイル。任意の名前で良い
```

現在slackbotディレクトリにいるとして、以下のコマンドを実行して、ファイルを作成する。

```
# mkdir -p /var/slackbot/plugins
# cd /var/slackbot
# touch run.py slackbot_settings.py plugins/__init__.py plugins/my_mention.py
```

5. slackbotの初期設定

(1) run.py の作成

run.py

```
-----
# coding: utf-8
from slackbot.bot import Bot

def main():
    bot = Bot()
    bot.run()

if __name__ == "__main__":
    print('start slackbot')
    main()
-----
```

(2) slackbot_settings.pyの作成

slackbot_settings.py

```
-----
# coding: utf-8

# botアカウントのトークンを指定
API_TOKEN = "xxxx-xxxxxxxxxxx-xxxxxxxxxxxxxxxxxxxxxxxx"

# このbot宛のメッセージで、どの応答にも当てはまらない場合の応答文字列
DEFAULT_REPLY = "何言ってんだこいつ"

# プラグインスクリプトを置いてあるサブディレクトリ名のリスト
PLUGINS = ["plugins"]
-----
```

6. slackbotの起動確認

以下のコマンドを入力して、プログラムを起動する。

```
$ python run.py
```

Slack上でbotへのダイレクトメッセージで何か投稿すると、デフォルトの返事が返ってきます。
botの参加しているチャンネルの場合は、メンションを付ける必要があります。

例

@agent メンションをつけると反応する

→ DEFAULT_REPLY が返ってくる

普通に投稿しても反応しない

→ 反応無し



shinoda 2:28 AM
joined channel_bot



shinoda 2:28 AM
set the channel purpose: チャットボット連携チャンネル



agent APP 2:28 AM
joined channel_bot by invitation from @shinoda



shinoda 2:37 AM
@agent メンションをつけると反応する



agent APP 2:37 AM
@shinoda: 何言ってんだこいつ



shinoda 2:37 AM
普通に投稿しても反応しない

7. slackbotへの機能追加

my_mention.pyを編集することで、機能を追加することができる。

7-1. ダイレクトメッセージやチャンネルの投稿を監視する

例として、以下を書き込む。

```
# coding: utf-8
from slackbot.bot import respond_to # @botname: で反応するデコーダ
from slackbot.bot import listen_to # チャンネル内発言で反応するデコーダ
from slackbot.bot import default_reply # 該当する応答がない場合に反応するデコーダ

@respond_to('メンション')
def mention_func(message):
    message.reply('私にメンションと言っているのだ') # メンション

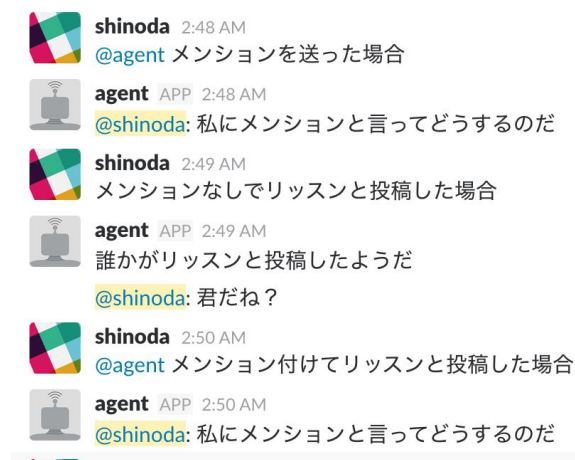
@listen_to('リッスン')
def listen_func(message):
    message.send('誰かがリッスンと投稿したようだ') # ただの投稿
    message.reply('君だね?') # メンション
```

<投稿テスト内容>

@agent メンションを送った場合

メンションなしでリッスンと投稿した場合

@agent メンション付けてリッスンと投稿した場合



```
# message.reply('string') @発言者名: string でメッセージを送信
# message.send('string') string を送信
# message.react('icon_emoji') 発言者のメッセージにリアクション(スタンプ)する
#                               文字列中に':'は入らない
```

slackbot の基本動作

- ・ @respond_to('hoge') デコレータを付けた関数
-> botに向けた投稿で、引数の文字列が含まれるときに反応する。
- ・ @listen_to('hoge') デコレータを付けた関数
-> チャンネル内の botに向けた投稿以外で、引数の文字列が含まれるときに反応する。
- ・ 反応する文字列は完全一致ではなく、含まれていればよい。正規表現が可能。

7-2. リアクション(スタンプ)をする

my_mention.pyにさらに以下を追加する。

```
@respond_to('かっこいい')
def cool_func(message):
    message.reply('ありがとう。スタンプ押しとくね') # メンション
    message.react('+1') # リアクション
```

<投稿テスト内容>

@agent かっこいいよ

7-3. デフォルトの返事をカスタマイズ

デフォルトの返事についてもプログラムで指定することができる。

my_mention.pyにさらに以下を追加する。

```
count = 0

@default_reply()
def default_func(message):
    global count # 外で定義した変数の値を変えられるようにする
    count += 1
    message.reply('%d 回目のデフォルトの返事です' % count) # メンション
```

<投稿テスト内容>

@agent 該当しない文字列を投稿

7-4. 反応する文字列を正規表現で指定

反応する文字列は正規表現で指定することができる。

my_mention.pyにさらに以下を追加する。

```
@respond_to(r'^ping\s+\d+\.\d+\.\d+\.\d+\s*$')
def ping_func(message):
    message.reply('それはpingのコマンドですね。実行できませんが') # メンション
```

<投稿テスト内容>

@agent ping 192.168.1.1

```
@listen_to(r'^篠田.*$')
def msgto_shinoda(message):
    message.reply('篠田は只今不在なので後ほど返答します。') # メンション
```

<投稿テスト内容>

篠田様 こんにちは

7-5. 受け取ったメッセージを取り出す

受け取ったメッセージを取り出す

slackbotは受け取ったメッセージを取り出すことができる。

my_mention.pyで先程書いたデフォルトの返事に関する部分を以下のように書き換える。

```
-----
@default_reply()
def default_func(message):
    text = message.body['text']    # メッセージを取り出す
    # 送信メッセージを作る。改行やトリプルバッククォートで囲む表現も可能
    msg = 'あなたの送ったメッセージは\n``'+ text + ''`'
    message.reply(msg)    # メンション
-----
```

<投稿テスト内容>

@agent こんにちは

7-6. 天気予報を返す

```
-----
@respond_to('天気')
def weather(message):
    import urllib
    import json

    url = 'http://weather.livedoor.com/forecast/webservice/json/v1?city='
    # '130010' とすると東京の情報を取得してくれる
    # ここを変えれば任意の地域の天気情報を取得できる
    city_id = '130010'

    html = urllib.request.urlopen(url+city_id)
    jsonfile = json.loads(html.read().decode('utf-8'))
    text = jsonfile['description']['text']
    message.send(text)
-----
```

<投稿テスト内容>

@agent 天気

7-7. 電車の遅延情報を返す

```
@respond_to('遅延')
def train_delay(message):
    import urllib
    import json
    uri = 'http://api.tetsudo.com/traffic/atom.xml?qshu'
    html = urllib.request.urlopen(uri+city_id)
    jsonfile = json.loads(html.read().decode('utf-8'))
    text = jsonfile['description']['text']
    message.send(text)
```

<投稿テスト内容>

@agent 遅延

8. slackbotの起動スクリプト

/var/slackbot/chatbot.sh

```
-----
#!/bin/bash
DAEMON=/var/slackbot/run.py
DAEMON_NAME=slackbot
PIDFILE=/var/run/$DAEMON_NAME.pid

log_daemon_msg() {
    msg=$1
    echo $msg
}

exec_start() {
    do_check 0
    running=$?
    if [ $running -eq 1 ]; then
        do_check 1
        return
    fi
    log_daemon_msg "Starting system $DAEMON_NAME daemon"
    python $DAEMON 2>&1>/dev/null &
    sleep 1
    pid=`ps aux|grep $DAEMON |grep -v grep|xargs echo|cut -d" " -f2`
    echo $pid > $PIDFILE
    do_check 1
}

exec_stop() {
    do_check 0
    running=$?
    if [ $running -eq 0 ]; then
        do_check 1
        return
    fi
    log_daemon_msg "Stopping system $DAEMON_NAME daemon"
    pid=`cat $PIDFILE`
    kill -9 $pid
    rm -f $PIDFILE
    do_check 1
}

do_check() {
    check=`ps aux|grep $DAEMON|grep -v grep|wc -l`
    show=$1
    if [ $show -eq 1 ]; then
        if [ $check -eq 0 ]; then
            log_daemon_msg "$DAEMON_NAME is not running."
        else
            log_daemon_msg "$DAEMON_NAME is running."
        fi
    fi
    return $check
}

case "$1" in

    start)
        exec_start
```

```
;;
stop)
    exec_stop
;;

restart|reload|force-reload)
    exec_stop
    exec_start
;;

status)
    do_check 1
;;
*)
    echo "Usage: /etc/init.d/$DAEMON_NAME {start|stop|restart|status}"
    exit 1
;;

esac
exit 0
```

9. slackbotのリマインダ機能

slash command

メッセージ入力で / から始まるコマンドを入力する事を slash command と言う。

- ・ irc や skype 等 chat アプリでは一般的な機能で， slack にも実装されている。
- ・ リマインダ機能 /remind も slash command の一つ。

登録方法

/remind [target] to [message] [at|in|on] [日時] (every day)

順序はわりと自由。上記の順でないといけないという事はない。

target	通知先の指定。 me 自分だけ。@here 投稿先のチャンネル。 #general チャンネル指定。 mention や チャンネル名もこの形式で指定。 @ で group mention だけでなく他人も指定できるので，他人のスケジュールを勝手に設定する事も可能
at in on 日時	通知したい時間を指定。 at は時間を指定，on は日にちを指定，in は n 分後/時間後等を指定。 ・in 15 minutes (15分後) ・at 6pm tomorrow (明日の午後6時に) ・on March 9th at 8:55pm
	繰り返しのイベントを作りたい場合 /remind me 猫写真をアップ on Tuesdays (毎週火曜日に) /remind me 犬写真をアップ at 11:00 every Thursday (毎週木曜 11:00に) /remind me 確定申告の準備 every January 25 (毎年1月25日に) /remind me 家賃の振込み！ on the 4th of every month (毎月4日に)

登録済の remind を確認する

/remind list

と打つと，自分が受け取るであろう通知が確認できる。

ヘルプ

/remind help

と入力すると下記のように slack bot が使い方を教えてくれる。

- ・`/remind me to drink water at 3pm every day`
- ・`/remind me on June 1st to wish Linda happy birthday`
- ・`/remind #team-alpha to update the project status every Monday at 9am`
- ・`/remind @jessica about the interview in 3 hours`
- ・`/remind @peter tomorrow "Please review the office seating plan"`

10. 株価情報を返す

jsm は日本の株式市場の株価・財務データを取得するツール。

Installation

```
# pip install jsm
```

```
/var/slackbot/plugins/stock_price.py
```

```
import jsm
import warnings

warnings.simplefilter("ignore")

def get_price(tag):
    ret = {}
    str1 = str(tag)
    elems = str1.split(',')
    for elem in elems:
        str2, str3 = elem.split("date:")
        date, str4 = str3.split(' 00:')
        str5, str6 = str4.split(" close:")
        close_price, _ = str6.split(" volume:")
        #print(date, close_price)
        ret[date] = close_price
    return ret

def get_stock_value(scode):
    strlist = []
    # インスタンス生成
    q = jsm.Quotes()
    # 当日の株価の取得
    price = q.get_price(scode).close
    strlist.append("---- code ----:" + str(scode) + "\r")
    strlist.append("---- 当日の株価 ----:" + str(price) + "\r" + "\r")

    # 過去の株価の取得
    # デフォルトでは当日から過去1ヶ月のデイリー株価を取得します。
    strlist.append("---- 過去の株価 ----:" + "\r")
    price_list = get_price(q.get_historical_prices(scode))
    for k,v in price_list.items():
        strlist.append(k + ":" + v + "\r")
    ret = strlist
    return ret
```

@agent 株価 5901
のように聞いた場合のコード

```
@respond_to(r'^株価.*$')
def stock_price(message):
    from stock_price import get_stock_value
    text = message.body['text'] # メッセージを取り出す
    try:
        _scode = text.split("株価 ")
    except ValueError as err:
        scode = ""
    if scode.isdigit():
        scode = scode.strip()
        strlist = get_stock_value(scode)
        ret = ""
        for v in strlist:
            ret = ret + str(v)
    else:
        ret = "知りたいコード番号と一緒に聞いてね"
    message.send(ret)
```

参考文献

PythonでSlackbotを作る(1)

<http://blog.bitmeister.jp/?p=3892>

PythonのslackbotライブラリでSlackボットを作る

<http://qiita.com/sukesuke/items/1ac92251def87357fdf6>

pythonを使って slackbotで株価をチェックする

<http://qiita.com/msrks/items/722bc2dc034fe5388861>

jsm 日本の株式市場の株価・財務データを取得するツール

<https://pypi.python.org/pypi/jsm/0.19>

リマインダーを設定する

<https://goo.gl/Dq85ms>

Beautiful Soup Documentation

<https://www.crummy.com/software/BeautifulSoup/bs4/doc/>