

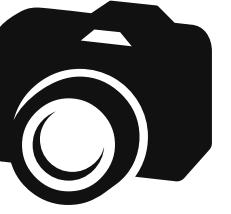
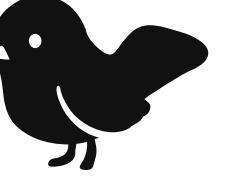
Automate the Boring Stuff with Slackbot

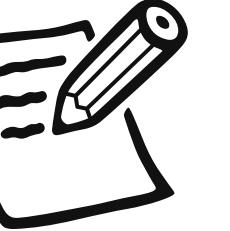
Takanori Suzuki

PyCon Malaysia / 2019 Aug 24

Agenda

- Background and Motivation for Slackbot
- How to create SIMPLE chatbot
- How to create INTERACTIVE bot
- How to EXTEND bot using libs and APIs

Photos  **Tweets**  

Notes  

#pyconmy2019

Who am I?

- Takanori Suzuki / 鈴木 たかのり
- Twitter: [@takanory](#)
- Vice-Chair of PyCon JP Committee: [#pyconjp](#)
- Director of [BeProud Inc.](#)
- Organizer of [Python mini Hack-a-thon](#): [#pyhack](#)
- Captain of [Python Bouldering Club](#): [#kabepy](#)



PyCon JP 2019

- pycon.jp/2019
- Date: 2019 September 14(Sat)-17(Tue)
- Place: Tokyo, Japan



PyCon Tour 2019



- Feb: PyCon APAC **P H**
- May: US PyCon The icon is a stylized representation of the United States flag, featuring horizontal stripes and stars.
- Jun: PyCon Thailand **T H**
- Jul: EuroPython **C H**
- Aug: PyCon Malaysia **M Y** The icon is a stylized representation of the Malaysian flag, featuring vertical stripes of red, white, and blue.
- Aug: PyCon Malaysia **M Y** Now!!!
- Sep: PyCon Japan The icon is a stylized representation of the Japanese flag, featuring vertical stripes of red, white, and blue.
- Sep: PyCon Taiwan **T W**
- Oct: PyCon Singapore **S G**

Background and Motivation

Conference Tasks

- I held PyCon JP(2014-2016) as Chair
- Conference tasks:
 -  Keynotes, Talks and Trainings arrangement
 -  Ticket sales and reception
 -  Venue and facility(WiFi, Video...) management
 -  Foods,  Coffee,  Snacks and  Beers

Staff ask me the same things

- 40+ staff
-  NEW staff :  OLD staff = 50 : 50

Programmer is Lazy

Let's create a secretary!!

Goal

- How to create SIMPLE bot
- How to create INTERACTIVE bot
- How to EXTEND bot using libs and APIs

Why Slackbot

- Launching the Slack app at any time  
- Easy to access Slack
- To do everything in Slack

#pycon-overseas ★
☆ 5 | 世界のPyCon情報 <https://tinyurl.com/r54v7w4>  106  

 **takanory**  00:13
20分からじゃねーのかよ!!! (edited) Saturday, May 15th ▾ 

 **terapyon** 00:13
画像の話?

 **canzawa** 00:13
ですねw

 **canzawa** 00:18
取り敢えず啓蒙?しました。
<https://twitter.com/yutakanzawa/status/1393224062713798656>

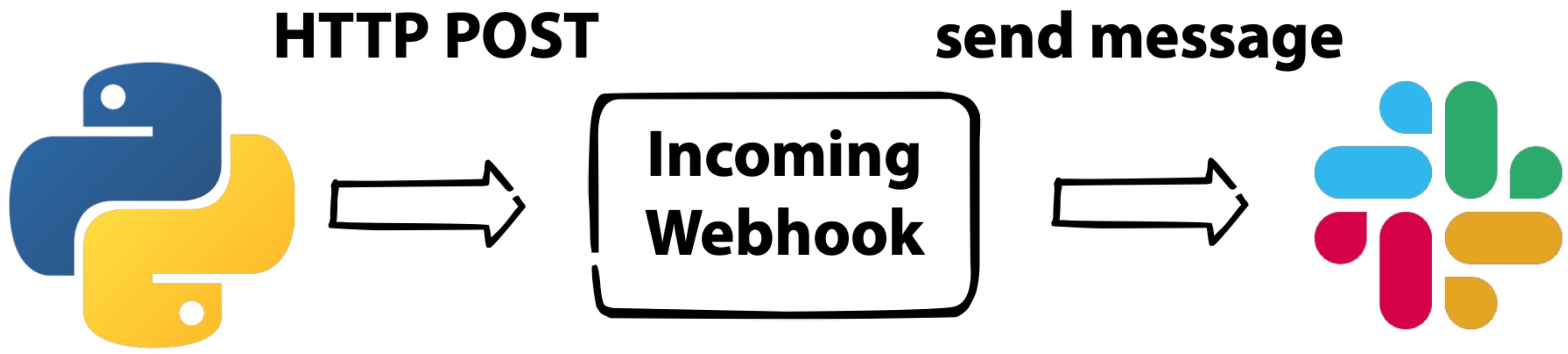
 **Yuta Kanzawa** @yutakanzawa

Keynote of #PyConUS now! #PyCon

<https://pbs.twimg.com/media/E1W5uszVUAEP589.jpg>

Simple integration with Incoming Webhooks

System overview



 mybot APP 19:14
Hello Slack
Hello Slack from Python!

Create Incoming Webhooks Integration

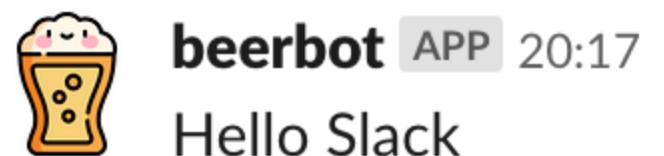
- Generate Webhook URL
 - 1. Create a Slack app
 - 2. Enable Incoming Webhooks in the app
 - 3. Create an Incoming Webhook
- Webhook URL:

<https://hooks.slack.com/services/T00000000/B00000000/XXXXXXXXXX>

- see: [Incoming Webhooks](#)

Post a message with cURL

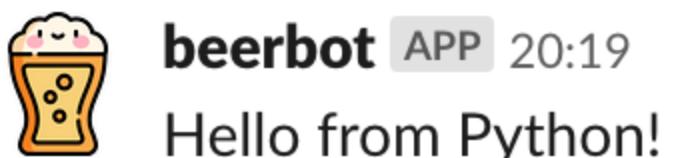
```
$ curl -X POST -H 'Content-type: application/json' \
> --data '{"text": "Hello Slack"}' \
> https://hooks.slack.com/services/T00000000/B00000000/XXXXXXX
```



Post a message with Python

```
import json
from urllib import request

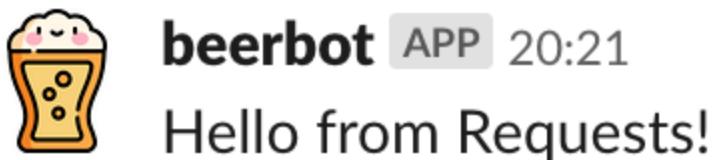
URL = 'https://hooks.slack.com/services/T00000000/B00000000/XXX'
data = {'text': 'Hello from Python!'}
jsoned = json.dumps(data).encode('utf-8')
req = request.Request(URL, data=jsoned, method='POST')
```



Post a message with Requests

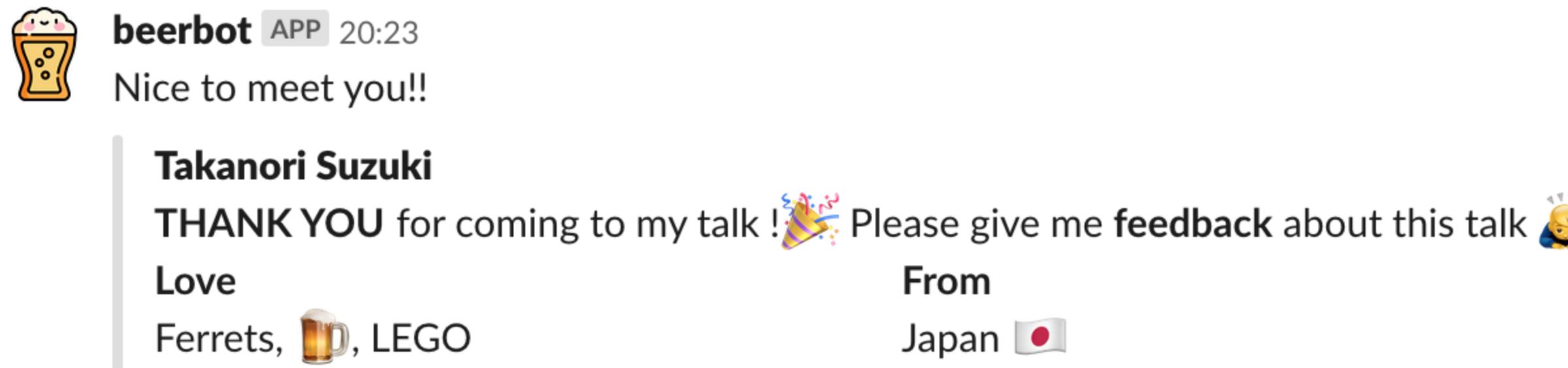
```
import requests
```

```
URL = 'https://hooks.slack.com/services/T00000000/B00000000/XXX'
data = {'text': 'Hello from Requests!'}
requests.post(URL, json=data)
```



Complex message with Requests

```
fields = [ {'title': 'Love', 'value': 'Ferrets, :beer:, LEGO', 'short': True},  
          {'title': 'From', 'value': 'Japan :jp:', 'short': True} ]  
  
data = { 'attachments': [ {  
    'pretext': 'Nice to meet you!!',  
    'author_name': 'Takanori Suzuki',  
    'author_link': 'https://twitter.com/takanory/',  
    'text': '*THANK YOU* for coming to my talk !:tada: Please give me feedback about this talk !:partying_face:',  
    'image_url': 'https://i.imgur.com/1234567890.jpg',  
    'color': '#36A2EB' } ] }
```



More complex message

- Block-Kit: new UI framework
- [Introducing Block Kit](#)
- [Block Kit Builder](#)

T Section

Image

Context

Divider

Actions

Section with...

Image

Button

Select

Multi-Select

Overflow

Datepicker

Fields

Message Preview ▾

Desktop ▾

Select a Template

 Your App APP 2:37 AM

Hello, Assistant to the Regional Manager Dwight! **Michael Scott** wants to know where you'd like to take the Paper Company investors to dinner tonight.

Please select a restaurant:

Farmhouse Thai Cuisine

 1528 reviews

They do have some vegan options, like the roti and curry, plus they have a ton of salad stuff and noodles can be ordered without meat!! They have something for everyone here



Kin Khao

 1638 reviews

The sticky rice also goes wonderfully with the caramelized pork belly, which is absolutely melt-in-your-mouth and so soft.



Ler Ros

 2082 reviews

I would really recommend the Yum Koh Moo Yang - Spicy lime dressing and roasted quick marinated pork shoulder, basil leaves, chili & rice powder.



[Farmhouse](#)

[Kin Khao](#)

[Ler Ros](#)

```

1 { "blocks": [
2   {
3     "type": "section",
4     "text": {
5       "type": "mrkdwn",
6       "text": "Hello, Assistant to the Regional Manager Dwight! *Michael Scott* wants to
7       know where you'd like to take the Paper Company investors to dinner tonight.\n\n *Please select a
8       restaurant:*

```

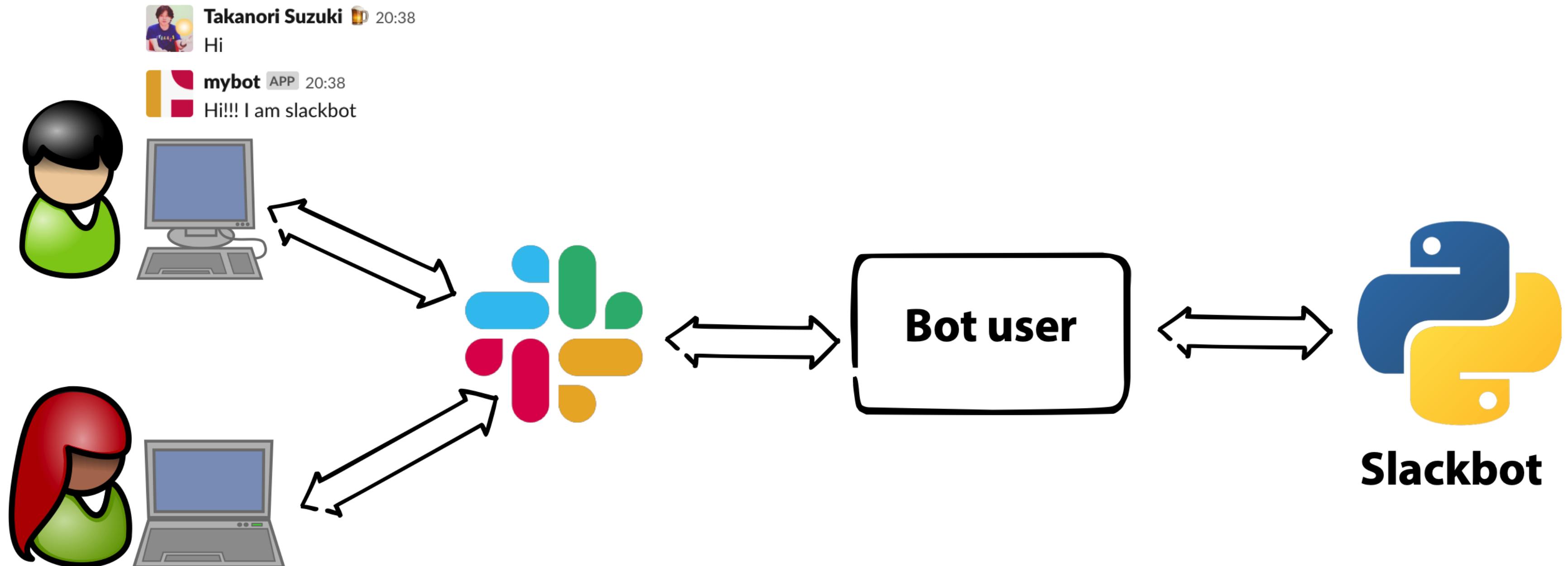
@takanory

Summary of Incoming Webhooks

- EASY to send messages from programs
- We can create COMPLEX messages
- But ONE WAY only(program -> Webhook -> Slack)

How to create slackbot

System overview



Create bot user on Slack

- Create bot user 
- 1. Create a Slack app
- 2. Enable Bots
- 3. Add a "Bot User"
- 4. Install App to Workspace -> Authorize
- "Bot User OAuth Access Token": **xoxb-123467890-XXXXXX-XXXXXXXXXXXX**
- Invite Bot User to Slack channels
- see: [Creating a bot user](#)

Install slackbot package

```
$ mkdir mybot; cd mybot  
$ python3.7 -m venv env  
$ . env/bin/activate  
(env) $ pip install slackbot
```

- github.com/lins05/slackbot

Create a simple bot with slackbot

- `slackbot_settings.py`

```
API_TOKEN = "xoxb-123467890-XXXXXX-XXXXXXXXXXXX" # Bot token
PLUGINS = ['mybot.plugins'] # Plugin packages
```

- `run.py`

```
from slackbot.bot import Bot
```

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

```
if __name__ == "__main__":
```

Simple Plugin

- mybot/plugins/__init__.py

```
(env) $ mkdir mybot
(env) $ mkdir mybot/plugins
(env) $ touch mybot/plugins/__init__.py # empty file
```

- mybot/plugins/sample.py

```
from slackbot.bot import listen_to
```

```
@listen_to('Hi')
def hello(message):
    message.send('Hi!!! I am slackbot')
```

File structure

File	Description
mybot/	Project dir
mybot/env/	venv
mybot/run.py	Main script
mybot/slackbot_settings.py	Settings
mybot/plugins/__init__.py	
mybot/plugins/sample.py	Sample plugin

Run slackbot

- (env) \$ python run.py



Takanori Suzuki 20:54

Hi



beerbot APP 20:54

Hi!!! I am beerbot!

Extend slackbot

listen_to / respond_to decorator

```
from slackbot.bot import listen_to, respond_to
```

```
@listen_to('Hi')
def hello(message):
    message.send('Hi!!! I am slackbot')
```

```
@respond_to('ping') # mention
```



Takanori Suzuki 14:40

Hi



beerbot APP 14:40

Hi!!! I am beerbot! 



Takanori Suzuki 14:40

cheers

@beerbot cheers!



beerbot APP 14:40

Cheers! 

@takanory

emoji reaction

- `message.react()` method

```
@listen_to('beer')
def beer(message):
    message.react(':beer:')
```



Takanori Suzuki 15:55

I want to drink craft beer in Pittsburgh!!!



Extract parameters on message

- Use regular expressions

```
@respond_to('choice (.*)') # choice pizza beer sushi
def choice(message, words): # -> words='pizza beer sushi'
    word = random.choice(words.split())
    message.send('I chose *{}*'.format(word))
```

Extract parameters on message



Takanori Suzuki 15:17

@beerbot choice pizza ramen sushi



beerbot APP 15:17

I chose sushi



Takanori Suzuki 15:17

@beerbot choice beer wine sake



beerbot APP 15:17

I chose beer



Takanori Suzuki 15:18

3 beers



beerbot APP 15:18



Takanori Suzuki 15:18

100beers



beerbot APP 15:18



Settings: slackbot_settings.py

```
ALIASES = '$' # Prefix instead of mention (@mybot ping -> $pin  
ERRORS_TO = 'mybot-error' # Error reporting channel  
DEFAULT_REPLY = "Sorry but I didn't understand you"  
PLUGIN = ['mybot.plugins', 'other.plugins',] # Pluing packages
```

-  **Takanori Suzuki** 15:31
\$choice beer wine sake
-  **beerbot** APP 15:31
I chose beer
-  **Takanori Suzuki** 15:32
\$ping
-  **beerbot** APP 15:32
@takanory: Sorry but I didn't understand you

- see: Usage of Slackbot

Attachments support

```
import json

@respond_to('follow me')
def followme(message):
    attachments = [{  
        'author_name': 'Takanori Suzuki',  
        'text': 'Follow me! <https://twitter.com/takanory> @taka  
    }]
```



Takanori Suzuki 15:34

\$follow me



beerbot APP 15:34

Takanori Suzuki

Follow me! @takanory

Summary of Slackbot

- We can COMMUNICATE with Slackbot
- Slackbot can handle ARGUMENTS in messages
- Slackbot can send messages in VARIOUS formats

Case studies

Calculator function using SymPy

Calculator function using SymPy

- Motivation
 - I feel heavy to call a calculator app on my smartphone
 - It seems useful if Slack as a calculator

Install SymPy

- SymPy: Python library for symbolic mathematics
 - www.sympy.org
- Install:
 - `$ pip install sympy`

- mybot/plugins/calc.py

```
from slackbot.bot import listen_to
from sympy import sympify, SympifyError

@listen_to(r'^([-*/^%!().\d\s]+)$') # Formula like pattern
def calc(message, formula):
    try:
        result = sympify(formula) # Simplifies the formula
        - -
    
```



Takanori Suzuki 15:45

1 + 1



beerbot APP 15:45

2



Takanori Suzuki 15:45

9500 / 3



beerbot APP 15:45

3,166.666666666666



Takanori Suzuki 15:45

$10^2 + 11^2 + 12^2$



beerbot APP 15:45

365



Takanori Suzuki 15:45

10!



beerbot APP 15:45

3,628,800

Plusplus function using Peewee ORM

Plusplus function using Peewee ORM

- Motivation
 - In PyCon JP, I want to make a culture that appreciates each other staff 

Install Peewee

- Simple and small ORM.
 - a small, expressive ORM
 - python 2.7+ and 3.4+ (developed with 3.6)
 - supports sqlite, mysql and postgresql
- docs.peewee-orm.com
- Install:
 - `$ pip install peewee`

Plusplus model

```
from pathlib import Path
from peewee import SqliteDatabase, Model, CharField, IntegerField

dbpath = Path(__file__).parent / 'plusplus.db' # db file
db = SqliteDatabase(dbpath)

class Plusplus(Model):
    pass
```

Plusplus command

```
from slackbot.bot import listen_to

from .plusplus_model import Plusplus

@listen_to(r'^(\w+)\+\+')
def plusplus(message, name):
    target = name.lower()
```



Takanori Suzuki 15:56

staff++



beerbot APP 15:56

Thank you staff! (count: 1)



Takanori Suzuki 15:56

Staff++



beerbot APP 15:56

Thank you Staff! (count: 2)



Takanori Suzuki 15:56

beer++



beerbot APP 15:56

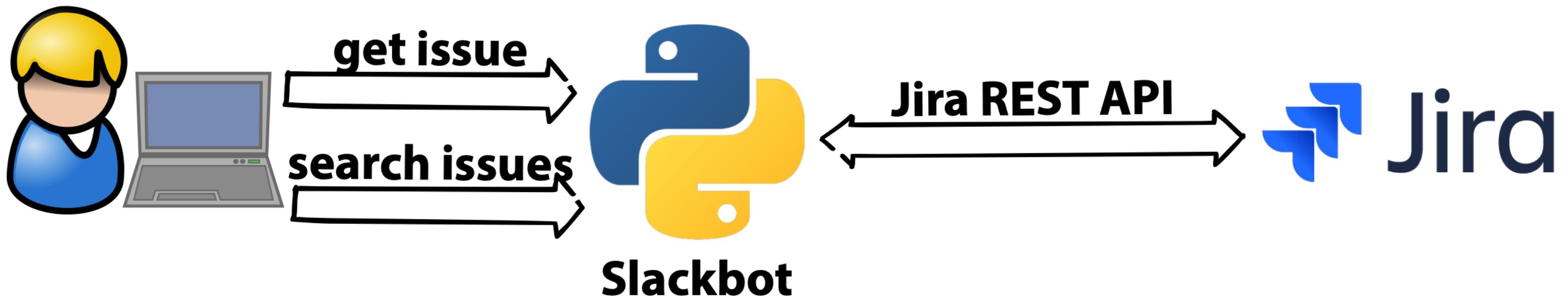
Thank you beer! (count: 1)

Display JIRA issue and Search issues

Display JIRA issue and Search issues

- Motivation
 - JIRA is very useful
 - JIRA Web is Very heavy
 - I want to check issue details without JIRA Web

System Overview



Install Python JIRA

- Python library to work with JIRA APIs
 - jira.readthedocs.io
- Install:
 - `$ pip install jira`

Authentication

```
from jira import JIRA
```

```
URL = 'https://jira.atlassian.com/'  
jira = JIRA(URL, basic_auth=('user', 'pass'))
```

- see: [2.1.2. Authentication](#)

Get Issue object

```
@listen_to(r'(ISSHA-[\d]+)') # Issue id pattern
def jira_issue(message, issue_id):
    issue = jira.issue(issue_id)
    summary = issue.fields.summary
    status = issue.fields.status.name
    assignee = issue.fields.assignee.name
    issue_url = issue.permalink()
```

- see: [2.1.3 Issues](#)



Takanori Suzuki 🍺 11:06

ISSHA-1484



mybot APP 11:06

ISSHA-1484 bot: amesh command

Assignee

takanory

Status

オープン

@takanory

Search issues

```
@respond_to('jira (.*)')
def jira_search(message, keywords):
    # make JQL query string
    jql = f'project=ISSHA and text ~ "{keywords}" order by crea
    text = ''
    # get 5 recent issues
    for issue in jira.search_issues(jql, maxResults=5):
        - - - - -
        • see: 2.1.5 Searching
        • JQL: JIRA Query Language
        • see: Advanced searching - Atlassian Documentation
```



Takanori Suzuki 🍺 19:31

\$jira pycon apac



mybot APP 19:31

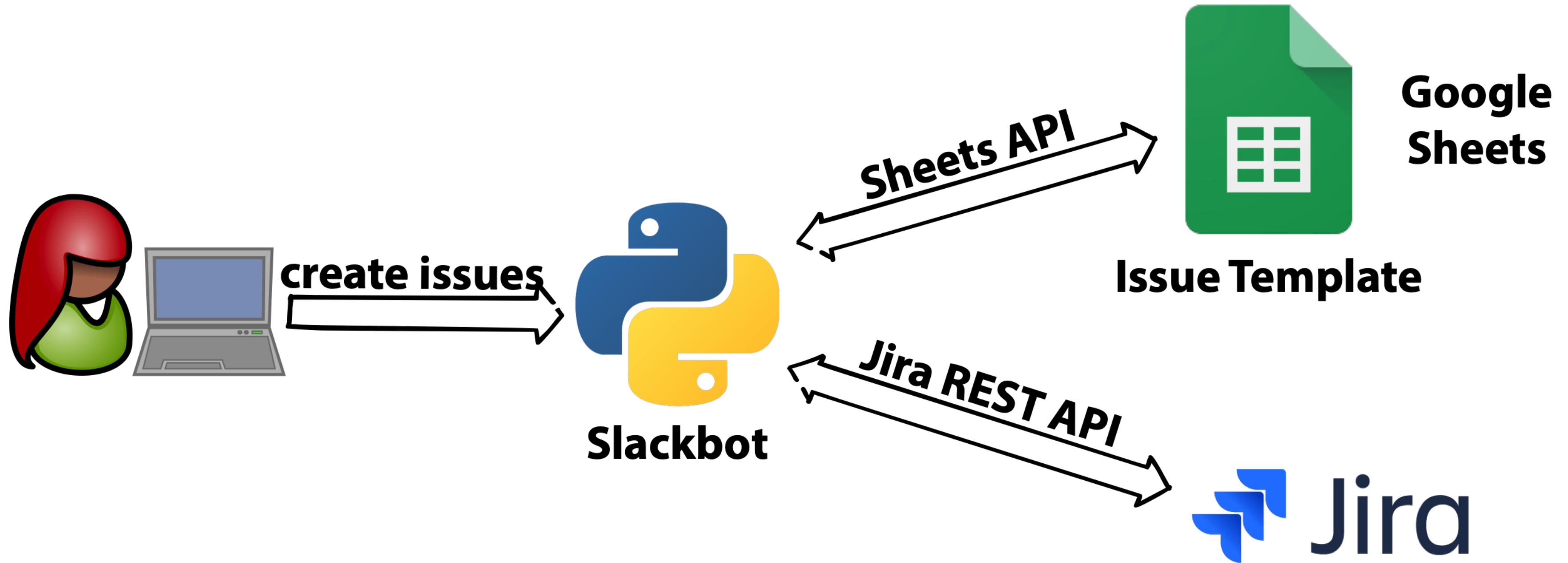
- [ISSHA-1549 OSC沖縄申込み](#)
- [ISSHA-1457 運営会議#26 \(2018年12月\) の準備](#)
- [ISSHA-1301 PyCon JP 2019 座長募集を行う](#)
- [ISSHA-1031 PyCon APAC 2018 シンガポール参加ツアーの段取り](#)
- [ISSHA-1024 一般社団法人PyCon JPの2017年の会計について報告する](#)

Create multiple issues from a template

Create multiple issues from a template

- Motivation
 - In pycamp event, **20+ issues** are required for each event
 - It is very painful for me to **copy issues** manually
 - JIRA Web is Very Heavy(again)

System Overview



Google Authorization is VERY Complex(1/2)

- create Google Cloud Platform project
 - enable API(in this case: Google Sheets API)
 - download `credentials.json`
- Install Google Client Library

```
$ pip install google-api-python-client google-auth-oauthlib
```

Google Authorization is VERY Complex(2/2)

- download [quickstart.py](#) on GitHub
- run `quickstart.py`
 - select your Google account in Web browser
 - Click the Accept button
 - get `token.pickle` file(finish!!)

(env) \$ python quickstart.py

Please visit this URL to authorize this application: <https://ac>

Name, Major:

Alexandra, English

:

- see: [Python Quickstart | Sheets API | Google Developers](#)

Get Spreadsheet Data

```
from googleapiclient.discovery import build  
SHEET_ID = '1LEtpNewhAFSXXXXXXXXXXXX'
```

```
creds = None  
if os.path.exists('token.pickle'):  
    with open('token.pickle', 'rb') as token:  
        creds = pickle.load(token)
```

- see: [Method: spreadsheets.values.get](#)

Create JIRA Issue

```
duedate = datetime.date.today + datetime.timedelta(days + 14)
issue_dict = {
    'project': {'key': PROJECT},
    'components': [{'name': COMPONENT}],
    'summary': SUMMARY_TEXT,
    'description': LONG_DESCRIPTION,
    'assignee': {'name': NAME_OF_ASSIGNEE},
```

- see: [2.1.3 Issues](#)

Sample of template command

- `$pycamp create`: Create issues for pycamp event

Place **Target date** **Staff name**

 **Tomohiro Kobayashi** 21:58
\$pycamp create 和歌山 4/20 kobatomo ottidododo massa142

 **pyconjpbot APP** 22:00
チケットを作成しました: <https://pyconjp.atlassian.net/browse/ISSHA-1521>

Bot created parent issue

pycampチケットテンプレート

ファイル 編集 表示 挿入 表示形式 データ ツール アドオン ヘルプ 最終編集: 2018年11月27日 (Ryuji Tsutsui さん)

Due date

Due date				
Category	Assignee	Summary	Description	
カテゴリー	担当者	日付	タイトル	本文
1.事前準備	コアスタッフ	-30	connpassイベント公開(現地スタッフ)	<p>h2. 目的</p> <ul style="list-style-type: none"> * 参加者を募るために、イベント本体、懇親会のconnpassイベントを作成して公開する * イベント開催の一ヶ月前くらいには公開したい <p>h2. 内容</p> <ul style="list-style-type: none"> * connpassイベントを過去イベントからコピーしてベースを作成(コアスタッフが実施) ** タイトルを変更する ** 日時を変更する ** 参加人数を消す ** 場所を未設定状態にする ** ロゴ画像を削除する ** アンケートを過去イベントからコピーする * 現地スタッフは以下を記述 ** ロゴ設定 ** 会場設定 ** 説明文の修正(場所、講師など) * 公開前にコアスタッフに内容のレビュー依頼をSlackでする * 公開のタイミングは平日の昼間が狙い目（多くの人に見てもらえる） * 懇親会のconnpassは場所未定で公開でもOKです。公開後に早めに場所を決めてください <p>以下、イベント公開に関するマニュアルも参照してください。 http://bootcamp-text.readthedocs.io/organize/1_manual.html#id8</p>
1.事前準備	講師	-30	ホテル、移動の手配(講師)	<p>h2. 目的</p> <ul style="list-style-type: none"> * 現地往復の交通手段を事前に予約する * ホテルの予約をする <p>h2. 内容</p> <ul style="list-style-type: none"> * 予約した内容と金額の記載をする * 支払いは、Python Boot Camp 終了後に別チケットで行います
1.事前準備	コアスタッフ	-21	事前打ち合わせ(主: コアスタッフ)	<p>h2. 目的</p> <ul style="list-style-type: none"> * 現地スタッフ、コアスタッフ、講師のやること認識合わせのために事前打ち合わせを実施する * Slack の Call を使う想定 <p>h2. 内容</p> <ul style="list-style-type: none"> * pyconjp-fellow.slack.comに#地域名（県単位）のチャンネルを作っておく * 作ったらpyconjp.slack.comの#pycampで現地スタッフ・TA・講師に報告する

Python Boot Camp in 和歌山を開催

編集	コメント	割り当て	処理中止	課題の解決	課題のクローズ	管理 ▾
タイプ:	<input checked="" type="checkbox"/> タスク	ステータス:	進行中 (ワークフローを表示)			
優先度:	重度	解決状況:	未解決			
コンポーネント:	Python Boot Camp					
ラベル:	なし					

説明

目的

- Python Boot Camp in 和歌山を開催するのに必要なタスクとか情報をまとめる親タスク

内容

- 日時: 2019-04-20(Sat)
- 会場: T-LABO
- 現地スタッフ: 落合大輔
- 講師: [Masataka Arai](#)
- TA: 堀口日向, Shimasan
- イベントconnpass: <https://pyconjp.connpass.com/event/120116/>
- 懇親会connpass: <https://pyconjp.connpass.com/event/120117/>

JIRAの使い方については以下マニュアルを参照してください。

http://bootcamp-text.readthedocs.io/organize/_1_manual.html#id28

1.事前準備

- [ISSHA-1522](#) CLOSED connpassイベント公開(現地スタッフ)
- [ISSHA-1523](#) OPEN ホテル、移動の手配(講師)
- [ISSHA-1524](#) IN PROGRESS 事前打ち合わせ(主: コアスタッフ)

2.広報

- [ISSHA-1525](#) CLOSED 事前ブログ執筆(現地スタッフ)
- [ISSHA-1526](#) IN PROGRESS メディアスポンサー経由での告知(コアスタッフ)
- [ISSHA-1527](#) CLOSED Twitter定期通知(コアスタッフ)
- [ISSHA-1528](#) IN PROGRESS 現地メディア経由での告知(現地スタッフ)

3.イベント直前、当日

- [ISSHA-1529](#) OPEN 参加者への事前連絡(現地スタッフ)
- [ISSHA-1530](#) OPEN お茶、お菓子購入(現地スタッフ)
- [ISSHA-1531](#) OPEN 案内スライド作成(現地スタッフ)
- [ISSHA-1532](#) OPEN 座席表作成、印刷(現地スタッフ)
- [ISSHA-1533](#) IN PROGRESS ランチミーティング(主: 現地スタッフ)
- [ISSHA-1534](#) OPEN 参加者アンケート実施(現地スタッフ)

4.事後処理

- [ISSHA-1535](#) OPEN 精算処理(現地スタッフ)
- [ISSHA-1536](#) OPEN 精算処理(講師)
- [ISSHA-1537](#) OPEN 事後ブログ(現地スタッフ)

Source code of pycamp command

- see:

github.com/pyconjp/pyconjpbot/blob/master/pyconjpbot/plugins/pycamp.py

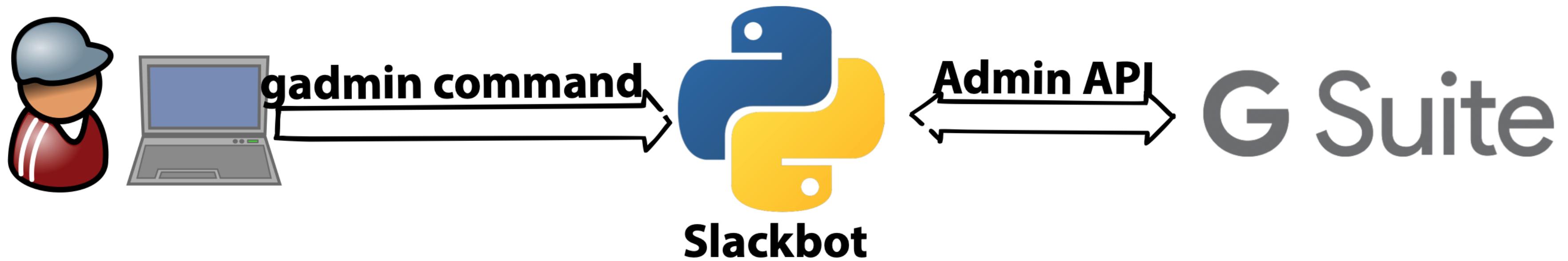
I never have to copy issues again 💥

Account management of G Suite

Account management of G Suite

- Motivation
 - PyCon JP use `pycon.jp` domain with G Suite
 - I only use Google Admin web occasionally
 - I forgot to use admin screen

System Overview



Update Google Authorization

- update Google Cloud Platform project
 - add G Suite Admin API
 - re-download `credentials.json`
- re-run `quickstart.py`
 - get new `token.pickle`

```
(env) $ python quickstart.py
```

Please visit this URL to authorize this application: <https://ac>

Name, Major:

Alexandra, English

:

Get user list

```
# build service
DOMAIN = 'pycon.jp'
service = build('admin', 'directory_v1', credentials=creds)

# get user list
users_list = service.users().list(orderBy='email', domain=DOMAIN)
for user in users_list.get('users', []):
    --snip--
```

- see: [Users: list | Directory API](#)

Insert user

```
body = {  
    'primaryEmail': EMAIL_ADDRESS,  
    'password': PASSWORD,  
    'name': {  
        'givenName': FIRST_NAME,  
        'familyName': LAST_NAME,  
    },  
}
```

- see: [Users: insert | Directory API](#)

Suspend, Resume, Delete user

```
suspend = {'suspended': True}
```

```
service.users().update(userKey=EMAIL, body=suspend).execute()
```

```
resume = {'suspended': False}
```

```
service.users().update(userKey=EMAIL, body=resume).execute()
```

```
service.users().delete(userKey=email).execute()
```

- see: [Users: update | Directory API](#)
- see: [Users: delete | Directory API](#)

Takanori Suzuki 🍺 11:25

\$gadmin help



pyconjpbot APP 11:25

ユーザー管理

- `$gadmin user list`: ユーザーの一覧を返す
- `$gadmin user insert (ユーザー) (名前) (名字)`: ユーザーを追加する
- `$gadmin user delete (ユーザー)`: ユーザーを削除する(停止中のみ削除可)
- `$gadmin user reset (ユーザー)`: ユーザーのパスワードをリセットする
- `$gadmin user suspend (ユーザー)`: ユーザーを停止する(停止中にする)
- `$gadmin user resume (ユーザー)`: ユーザーを再開する(アクティブにする)

メールのエイリアス管理

- `$gadmin alias list (ユーザ)`: ユーザーのエイリアスの一覧を返す
- `$gadmin alias insert (ユーザ) (エイリアス)`: ユーザーにエイリアスを追加する
- `$gadmin alias delete (ユーザ) (エイリアス)`: ユーザーからエイリアスを削除する

グループ管理

- `$gadmin group list`: グループの一覧を返す
- `$gadmin group insert (グループ) (グループ名)`: 指定したグループを追加する
- `$gadmin group delete (グループ)`: 指定したグループを削除する

グループのメンバー管理

- `$gadmin member list (グループ)`: 指定したグループのメンバー一覧を返す
- `$gadmin member insert (グループ) (メール1) [(メール2...)]`: 指定したグループにメンバーを追加する
- `$gadmin member delete (グループ) (メール1) [(メール2...)]`: 指定したグループからメンバーを削除する

Takanori Suzuki 🍺 11:27
\$gadmin user list



pyconjpbot APP 11:27
pycon.jp ドメインのユーザー一覧(39ユーザー)

- 2015@pycon.jp PyCon JP 2015事務局
- 2016@pycon.jp PyCon JP 2016
- 2017@pycon.jp PyCon JP 2017
- association@pycon.jp 一般社団法人会計
- event@pycon.jp イベント会計
- pycon.jp user admin
- ai@pycon.jp Ai
- aron.jp Masataka
- cc@pycon.jp JP P
- ha@pycon.jp YOSHIMOTO
- iai@pycon.jp la
- iq@pycon.jp Iqbal A
- kii@pycon.jp Makoto
- kiyoshi@pycon.jp Fumiaki
- ko@pycon.jp Tsuboi Kobayashi
- kota@pycon.jp Kao
- m@pycon.jp 関
- manoda@pycon.jp Manami
- matsu@pycon.jp Ry
- medya@pycon.jp メディア
- pr@pycon.jp プロ
- py@pycon.jp Book
- py@pycon.jp python
- ry@pycon.jp Kamiya
- shi@pycon.jp Shimizukawa
- sotaro@pycon.jp スタジオ
- su@pycon.jp Suyama
- sy@pycon.jp Aoi
- syn@pycon.jp V
- syu@pycon.jp Shu
- ta@pycon.jp Taku
- te@pycon.jp Yuya
- te@pycon.jp Mana
- ts@pycon.jp Mamil

@takanory

Takanori Suzuki 🍺 11:30

\$gadmin group list

pyconjpbot APP 11:30

pycon.jp ドメインのグループ一覧(21グループ)

- 20 [REDACTED]@pycon.jp 2015グループ(2ユーザー)
- 20 [REDACTED]@pycon.jp 2016グループ(9ユーザー)
- 20 [REDACTED]@pycon.jp 2017グループ(6ユーザー)
- 20 [REDACTED]day-staff@pycon.jp 2018当日スタッフDocs権限用(16ユーザー)
- 20 [REDACTED]staff@pycon.jp 2019コアスタッフDocs権限用(11ユーザー)
- acc [REDACTED]min@pycon.jp 一般社団法人会計責任者グループ(2ユーザー)
- acc [REDACTED]ociation-group@pycon.jp 一般社団法人会計用グループアドレス(5ユーザー)
- acc [REDACTED]ent-group@pycon.jp イベント会計用グループアドレス(3ユーザー)
- app [REDACTED]on.jp AppleDeveloperProgram管理(2ユーザー)
- boar [REDACTED]on.jp 法人理事会(7ユーザー)
- cor [REDACTED]up@pycon.jp contactグループ(4ユーザー)
- des [REDACTED]on.jp デザイン(5ユーザー)
- hiro [REDACTED]pycon.jp PyCon mini Hiroshima(1ユーザー)
- ma [REDACTED]it-group@pycon.jp 事務局グループ(8ユーザー)
- me [REDACTED]o@pycon.jp メディアグループ(6ユーザー)
- pro [REDACTED]oup@pycon.jp プログラムグループ(6ユーザー)
- pycamp-[REDACTED]@pycon.jp Python Boot Camp(4ユーザー)
- pycc [REDACTED]up@pycon.jp pyconjp@pycon.jp の転送用グループ(3ユーザー)
- spor [REDACTED]up@pycon.jp スポンサーグループ(4ユーザー)
- sym [REDACTED]group@pycon.jp 2016(以降)のWebsiteのアラート送付先(4ユーザー)
- venu [REDACTED]@pycon.jp 会場グループ(15ユーザー)

Takanori Suzuki 🍺 11:30

\$gadmin member list pycamp-[REDACTED]

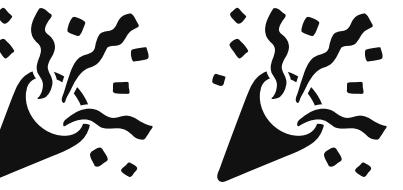
pyconjpbot APP 11:30

pycamp-[REDACTED]@pycon.jp グループのメンバー(4ユーザー)

- ko [REDACTED]@pycon.jp
- o6z [REDACTED]@pyconjp.slack.com
- ry [REDACTED]@gmail.com
- tak [REDACTED]@pycon.jp

@takanory

I can completely forget Google Admin web site



Source code of gadmin command

- see:

github.com/pyconjp/pyconjpbot/blob/master/pyconjpbot/google_plus.py

Summary

- Incoming Webhooks
- **slackbot**
- **slackbot with Libraries**
- **slackbot with APIs**

Next steps

- Let's make YOUR own Slackbot
- Let's connect with libraries and APIs
- Automate your Boring Stuff with Slackbot

Thank you! 🤲

Terima kasih! 🤲

- Twitter: [@takanory](https://twitter.com/takanory)
- Slides: github.com/takanory/slides