

RaspberryPiを使った定点カメラの作成

s1260242

Ryusei Takahashi

July,29

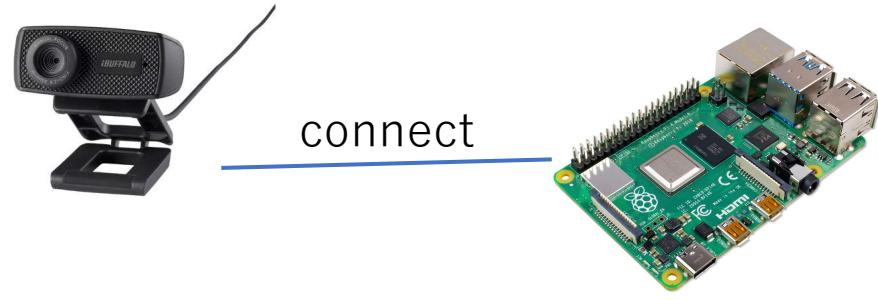
Contents

- Current Status before Today's class
- Motivation
- Purpose
- Functional spec
- Today's content
- What I did on Today's class
- Deliverable
- Schedule

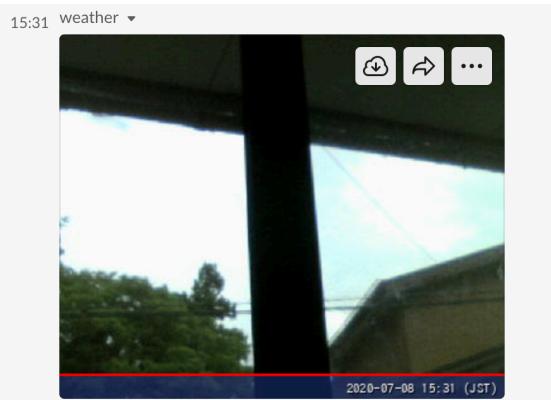
Current Status before Today's class

ラズパイに繋いだカメラで取得した画像をSlackにアップロードすることができる。

実装済みの機能



Send a picture by using
python code and Slack API



slack

実装済み

Slack APIのみ実装済み

単位時間ごとに写真を撮影し、
Slackに投稿できるように実装済み

Operation Flow

Take a picture
(RaspberryPi)

Get SlackAPI and Weather API
(PythonCode)

Send to general channel with text
on Slack
(Python)

Display picture and information
about weather per an hour
(Python Code)

Motivation

前回に引き続き、今日もOpenWeatherMapを用いて天気情報を取得できるようみたい。

(6/24に提出した計画スライドには、LivedoorのWeatherHackを利用すると記載していましたが、このサービスが7月末に終了してしまいますのでOpenWeatherMapを利用します。)

Purpose

- 定期的に自分のアパートの天気を確認して、帰省時や旅行時に使いたい。雨なら傘の準備ができ、気温の変化によって着ていくものを考えることができるようしたい。

Functional Spec

- カメラで写真を取得することができる。(capture.py)
- 取得した写真を保存し、Slackの指定したgeneralチャンネルに送信することができる。
- Slackに画像を送信する際、"That's the weather right now."と表示される。
- Crontab.textの設定によって単位時間あたりのプログラムの実行ができる。
- 三日間の5時間おきおの天気情報を取得できる。

Today's Content 2nd Period

前回作ったCode(rasWeather.py)を元に自分が必要な情報をSlackに出力するにはどうすれば良いかを模索しました。

OpenWeatherMapで年コードからその地域の情報が読み取れます、
会津若松市がその中に含まれていなかったので喜多方の天気情報を取得するようにしました。

また、slacker.chat.post_messageでは一文しか出力ができないので固定して表示したい文章と
変数として表示する文章を分けて表示するようにしました。

Today's Content 2nd Period (Completed Code)

```
API_KEY = 'da1bc6b2497e5f663ff00c738ed049f0'
city = "kitakata"
API_URL = 'http://api.openweathermap.org/data/2.5/weather?'
makeUrl = API_URL + "appid=" + API_KEY + "&q=" + city
response = requests.get(makeUrl)
cityData = response.json()

#For Slack
token = "xoxp-1234430567092-1213507232231-1228269505394-0a816f3d41448d3ff85fdcecb14c7e00"

slack = Slacker(token)
channel_name = "#" + "general"

# もし都市名が見つかりませんと404エラーが返る
if cityData["cod"] != "404":
    slack.chat.post_message("C016WF9UWN8", "Current Weather:", as_user=True)
    slack.chat.post_message("C016WF9UWN8", cityData["weather"][0]["description"], as_user=True)
    slack.chat.post_message("C016WF9UWN8", "Current Temperature()", as_user=True)
    slack.chat.post_message("C016WF9UWN8", round(cityData["main"]["temp"] - 273.15,1), as_user=True)
    slack.chat.post_message("C016WF9UWN8", "Maximum Temperature()", as_user=True)
    slack.chat.post_message("C016WF9UWN8", round(cityData["main"]["temp_max"] - 273.15,1), as_user=True)
    slack.chat.post_message("C016WF9UWN8", "Minimum Temperature()", as_user=True)
    slack.chat.post_message("C016WF9UWN8", round(cityData["main"]["temp_min"] - 273.15,1), as_user=True)
    slack.chat.post_message("C016WF9UWN8", "Humidity", as_user=True)
    slack.chat.post_message("C016WF9UWN8", cityData["main"]["humidity"], as_user=True)
    slack.chat.post_message("C016WF9UWN8", "Wind Speed", as_user=True)
    slack.chat.post_message("C016WF9UWN8", cityData["wind"]["speed"], as_user=True)
    slack.chat.post_message("C016WF9UWN8", "-----", as_user=True)
else:
    print("都市名がみつかりませんでした。")
```

Today's Content 3rd Period

天気情報を取得するPythonCodeは写真を撮るためのcapture.pyとは分けていたので、Slackに送信できるようにAPIなどの設定を行いました。

```
1 # -*- coding: utf-8 -*-
2 import json
3 import datetime
4 import os
5 import requests
6 import sys
7
8 from pytz import timezone
9 from slacker import Slacker
10
11 API_KEY = 'dalbc6b2497e5f663ff00c738ed049f0'
12 city = "kitakata"
13 API_URL = 'http://api.openweathermap.org/data/2.5/weather?'
14 makeUrl = API_URL + "appid=" + API_KEY + "&q=" + city
15 response = requests.get(makeUrl)
16 cityData = response.json()
17
18 #For Slack
19 token = "xoxp-1234430567092-1213507232231-1228269505394-0a816f3d41448d3ff85fdcecb14c7e00"
20
21 slacker = Slacker(token)
22 channel_name = "#" + "general"
23
24 # Codが404だと都市名が見つかりませんの意味
25 if cityData["cod"] != "404":
26     slacker.chat.post_message("C016WF9UWN8", "Current Weather:", as_user=True)
27     slacker.chat.post_message("C016WF9UWN8", cityData["weather"][0]["description"], as_user=True)
28     slacker.chat.post_message("C016WF9UWN8", "Current Temperature()", as_user=True)
29     slacker.chat.post_message("C016WF9UWN8", round(cityData["main"]["temp"] - 273.15,1), as_user=True)
30     slacker.chat.post_message("C016WF9UWN8", "Maximum Temperature()", as_user=True)
31     slacker.chat.post_message("C016WF9UWN8", round(cityData["main"]["temp_max"] - 273.15,1), as_user=True)
32     slacker.chat.post_message("C016WF9UWN8", "Minimum Temperature()", as_user=True)
33     slacker.chat.post_message("C016WF9UWN8", round(cityData["main"]["temp_max"] - 273.15,1), as_user=True)
34     slacker.chat.post_message("C016WF9UWN8", "Humidity", as_user=True)
35     slacker.chat.post_message("C016WF9UWN8", cityData["main"]["humidity"], as_user=True)
36     slacker.chat.post_message("C016WF9UWN8", "Wind Speed", as_user=True)
37     slacker.chat.post_message("C016WF9UWN8", cityData["wind"]["speed"], as_user=True)
38     slacker.chat.post_message("C016WF9UWN8", "-----", as_user=True)
39 else:
40     print("都市名がみつかりませんでした。")
```

Today's Content 3rd Period

Current Weather:

mist

Current Temperature(°C)

20.9

Maximum Temperature(°C)

21.1

Minimum Temperature(°C)

21.1

Humidity

100

Wind Speed

4.6

python3 rasWeather.pyを実行するとこのように自分が設定した出力をSlackで表示できるようになりました。

Today's Content 4th Period

```
*/1 * * * * python3 /home/pi/exercises/capture.py  
*/1 * * * * python3 /home/pi/exercises/rasWeather.py
```

とすることで1分ごとにSlackに送ることができるようになりました。

しかし、capture.pyとrasWeather.pyで実行時間が異なり次のページの画像のようにrasWeather.pyの出力の間に画像がくるような出力になってしまいます。

Today's Content 4th Period

Current Weather:

light intensity shower rain

Current Temperature(°C)

21.3

weather ▾



Maximum Temperature(°C)

21.7

Minimum Temperature(°C)

21.7

Humidity

100

#general へのメッセージ

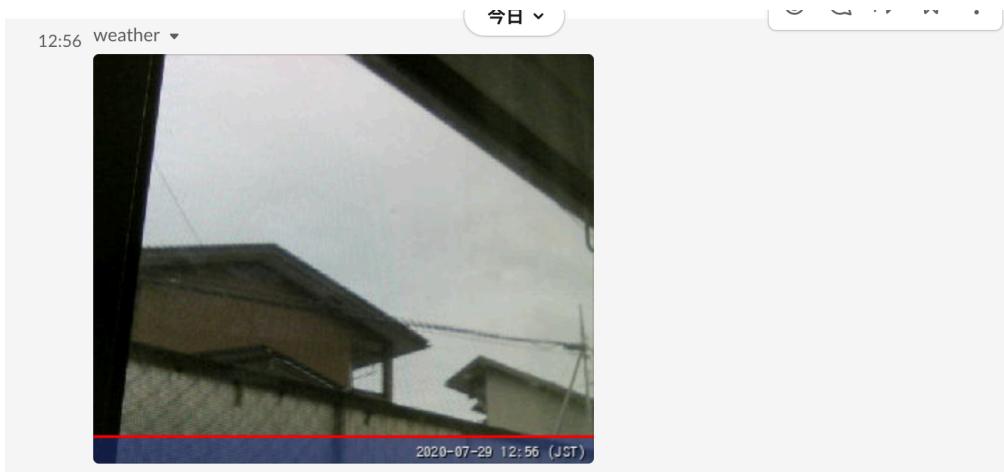
Today's Content 4th Period

実行時間の遅延を合わせるために、Ex4で学んでSleepを用いて実行の入れ子状態にならないように Crontab.txtを修正しました。

```
*/1 * * * * python3 /home/pi/exercises/capture.py  
*/1 * * * * sleep 2; python3 /home/pi/exercises/rasWeather.py
```

今回は2秒待つことで出力がしっかり分けることができます。

Today's Content 4th Period



しっかりと画像とその情報を分けることができた。

What I did on July 29

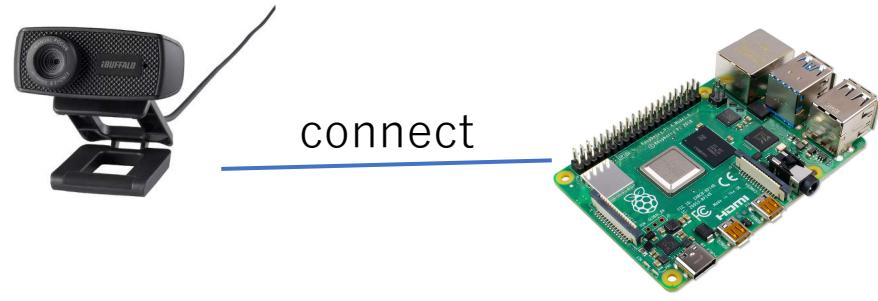
OpenWeatherMapを用いて、天気情報を取得し自分が出力したいデータを出力することができました。
左) capture.py 右) rasWeather.py

```
1 # -*- coding: utf-8 -*-
2 import os
3
4 os.system("fswebcam /dev/video0 /home/pi/exercises/weather.jpg")
5
6 from slacker import Slacker
7
8 #OAuthToken
9 token = "xoxp-1234430567092-1213507232231-1228269505394-0a816f3d41448d3ff85fdcecb14c7e00"
10
11 slacker = Slacker(token)
12 channel_name = "#" + "general"
13 result = slacker.files.upload("/home/pi/exercises/weather.jpg",channels=[ "C016WF9UWN8"])
14 #slacker.chat.post_message("C016WF9UWN8", "That's the weather right now.", as_user=True)
15 slacker.pins.add(channel="C016WF9UWN8", file_=result.body["file"]["id"])
16
```

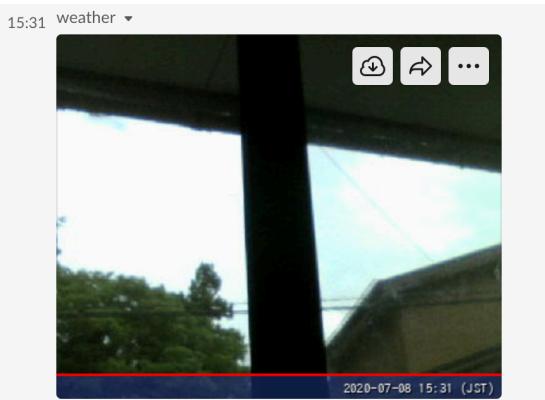
```
1 # -*- coding: utf-8 -*-
2 import json
3 import datetime
4 import os
5 import requests
6 import sys
7
8 from pytz import timezone
9 from slackr import Slackr
10
11 API_KEY = 'da1bc6b2497e5f663ff00c738ed049f0'
12 city = "kitakata"
13 API_URL = 'http://api.openweathermap.org/data/2.5/weather?'
14 makeUrl = API_URL + "appid=" + API_KEY + "&q=" + city
15 response = requests.get(makeUrl)
16 cityData = response.json()
17
18 #For Slack
19 token = "xoxp-1234430567092-1213507232231-1228269505394-0a816f3d41448d3ff85fdcecb14c7e00"
20
21 slacker = Slacker(token)
22 channel_name = "#" + "general"
23
24 # Codが404だと都市名が見つかりませんの意味
25 if cityData["cod"] != "404":
26     slacker.chat.post_message("C016WF9UWN8", "Current Weather:", as_user=True)
27     slacker.chat.post_message("C016WF9UWN8", cityData["weather"][0]["description"], as_user=True)
28     slacker.chat.post_message("C016WF9UWN8", "Current Temperature()", as_user=True)
29     slacker.chat.post_message("C016WF9UWN8", round(cityData["main"]["temp"] - 273.15,1), as_user=True)
30     slacker.chat.post_message("C016WF9UWN8", "Maximum Temperature()", as_user=True)
31     slacker.chat.post_message("C016WF9UWN8", round(cityData["main"]["temp_max"] - 273.15,1), as_user=True)
32     slacker.chat.post_message("C016WF9UWN8", "Minimum Temperature()", as_user=True)
33     slacker.chat.post_message("C016WF9UWN8", round(cityData["main"]["temp_min"] - 273.15,1), as_user=True)
34     slacker.chat.post_message("C016WF9UWN8", "Humidity", as_user=True)
35     slacker.chat.post_message("C016WF9UWN8", cityData["main"]["humidity"], as_user=True)
36     slacker.chat.post_message("C016WF9UWN8", "Wind Speed", as_user=True)
37     slacker.chat.post_message("C016WF9UWN8", cityData["wind"]["speed"], as_user=True)
38     slacker.chat.post_message("C016WF9UWN8", "-----", as_user=True)
39 else:
40     print("都市名がみつかりませんでした。")
```

What I did on July 29

実装済みの機能



Send a picture by using
python code and Slack API



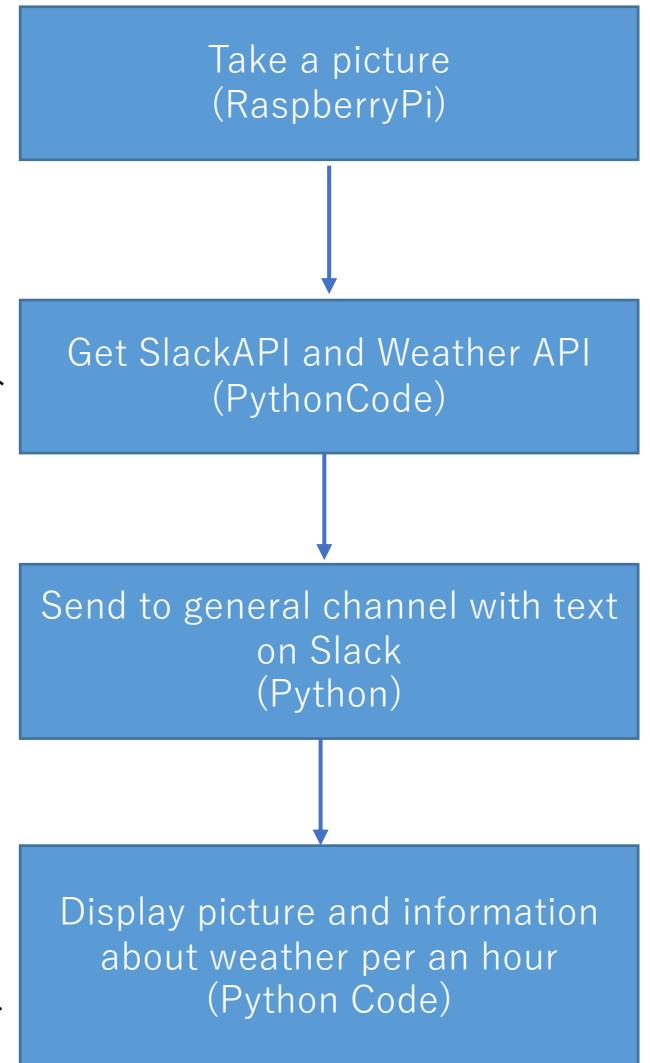
slack

単位時間ごとに写真を撮影し、
Slackに投稿できるように実装済み

実装済み

スラックAPIのみ実装済み
WeatherAPIの実装

Operation Flow



Deliverable

今回はCrontab.txtの実装により、1分おきに出力されるようになっている。



Schedule

- 7/8: カメラの動作確認とSlackのTokenの発行し、Pythonコードを作成して画像をアップロードできるようにする
- 7/15: crontabを用いて、1分おきに画像をアップロードできるようにする
- 7/22: 画像だけになってしまないので、日付と天気情報(気温や湿度)を表示する
- 7/29: 22日に同じ
- 8/4: プрезентーションの作成

8/4日にプレゼンテーションが作成できると思っていたのでこれから来週の火曜日までにプレゼンテーションを作成します。