SlackBot プログラムの仕様書 (修正版)

2020/6/24

野村 優文

1　本仕様書の修正箇所

(1)　「発言」と「投稿」という語句を混同して文中で使用していたので，使用する語句を「投稿」に統一した．

(2) Slack のサーバ，SlackBot プログラムの関係が文章のみの説明だと分かりにくい．このため，「SlackBot プログラムの処理の流れ」という章を作成し，図を挿入してこれらの関係を説明した．

(3) “” で括っている間に改行が含まれていたので，含まないように修正した．

(4) 5 章において，SlackBot プログラムの機能の説明が長くなっているため，表を作成して説明を短くした．

(5)「を」の後には読点をつけないように修正した．

(6) Heroku についての説明が無かったので，6 章の動作環境にて説明を追加した．

(7) 表 2 に書かれている動作環境が手元の PC の環境であったので，Heroku の動作環境に書き換えた．

(8) 9 章において，エラー処理の文章が長いため，箇条書きでまとめた．

2　概要

本資料は，2020 年度新人研修課題で作成した SlackBot プログラムの仕様をまとめたものである．本プログラムではチャットツールである Slack[1] を用いる．また，SlackBot は，ユーザが Slack 上で投稿した特定の文章をきっかけとして，Slack 上で自動的に返信する機能をもつ．本プログラムは以下の 2 つの機能をもつ．

(機能 1)　ユーザから指定された文字列を返信する機能

(機能 2)　天気予報の情報を返信する機能

3　対象とする利用者

本プログラムでは，Slack アカウントを所有する利用者を対象としている．

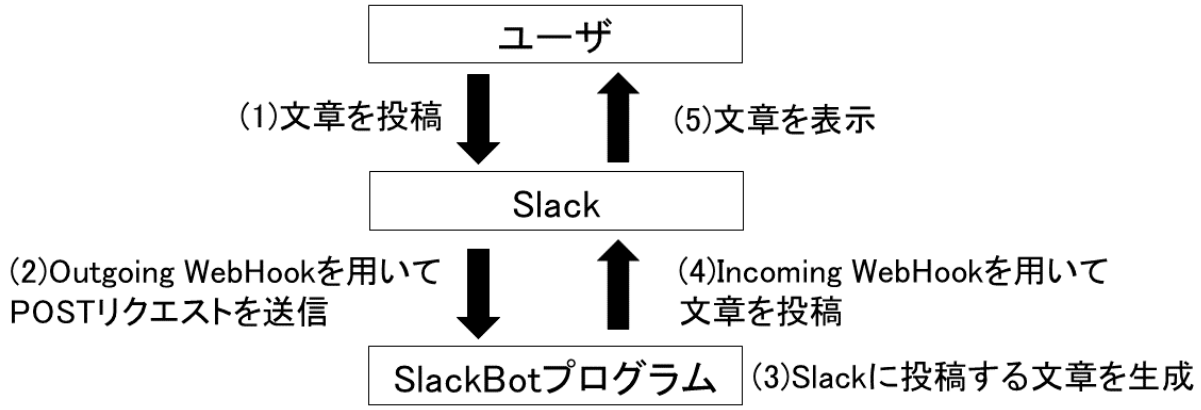

図 1　(機能 1) の処理の流れ

4　SlackBot プログラムの処理の流れ

4.1　(機能 1) の処理の流れ

(機能 1) の処理の流れを図 1 に示し，以下で各処理について説明する．

(1) ユーザが Slack へ文章を投稿する．
(2) Slack は Outgoing WebHook を用いて，SlackBot プログラムに POST リクエストを送信する．Outgoing WebHook とは，特定の文章が Slack に投稿されたとき，この特定の文章や投稿したユーザなどの情報を含んで，指定した URL に POST リクエストを送信する機能である．
(3) SlackBot プログラムは，Slack に投稿する文章を生成する．
(4) SlackBot プログラムは，Incoming WebHook を用いて，生成した文章を Slack に投稿する．Incoming WebHook とは，外部のサービスから Slack に文章を投稿する機能である．
(5) Slack は，SlackBot プログラムが投稿した文章を表示する．

4.2　(機能 2) の処理の流れ

(機能 2) の処理の流れを図 2 に示し，以下で各処理について説明する．

(1) ユーザが Slack へ文章を投稿する．

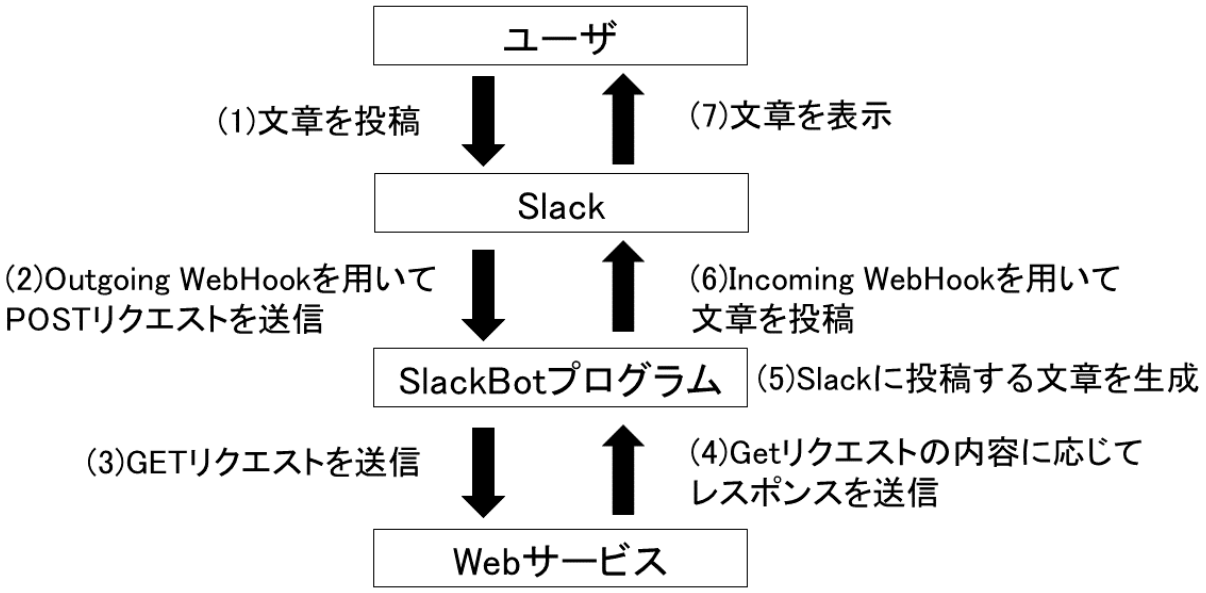

図 2　(機能 2) の処理の流れ

(2) Slack は Outgoing WebHook を用いて，SlackBot プログラムに POST リクエストを送信する．
(3) SlackBot プログラムは Web サービスに GET リクエストを送信する．
(4) Web サービスは，GET リクエストの内容に応じて，SlackBot プログラムにレスポンスを送信する．
(5) SlackBot プログラムは，Web サービスからのレスポンスを元にして，Slack に投稿する文章を生成する．
(6) SlackBot プログラムは，Incoming WebHook を用いて，生成した文章を Slack に投稿する．
(7) Slack は，SlackBot プログラムが投稿した文章を表示する．

5　機能

ユーザが Slack 上に “@masabot” で始まる文章を投稿したとき，本プログラムは受け取った文字列に対応する文章を Slack 上に返信する．本プログラムが返信する文章は，ユーザが投稿する “@masabot” の後ろに付く文字列によって決まる．以下に本プログラムがもつ機能について記述する．

(機能 1)　ユーザから指定された文字列を返信する機能

この機能は，ユーザが “@masabot 「(指定した文字列)」と言って” と投稿したとき，SlackBot プログラムは “「” と “」と言って” の間にある文字列を読み込んで，“(指定した文字列)” という文章を返信する．例えば，ユーザは “@masabot 「こんにちは」と言って” と投稿したとき，

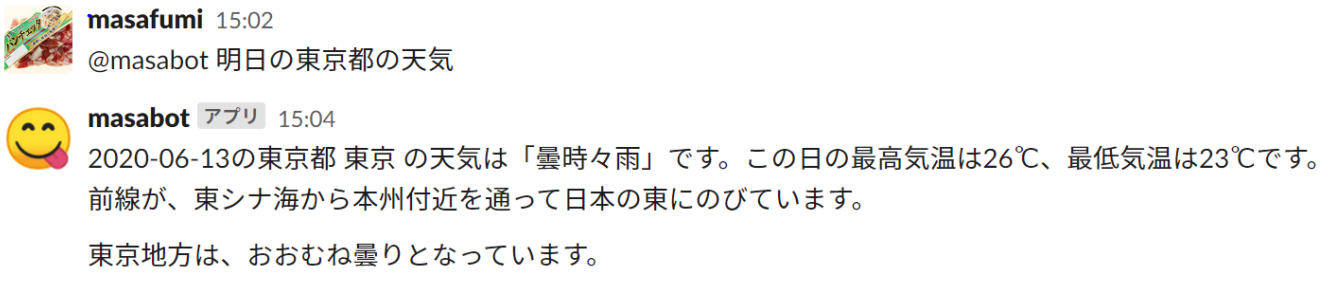

図 3　ユーザが “@masabot 明日の東京都の天気” と投稿したときの，SlackBot の返信

表 1　指定できる “(日にち)” と “(都道府県名)”

項目	内容	備考
“(日にち)”	“今日”，“明日”，“明後日”	
“(都道府県名)”	47 都道府県名のいずれか 1 つ	“都”，“府”，および “県” は省略可

Slackbot は “こんにちは” と返信する．ただし，“(指定した文字列)” の中に “「”，もしくは “」” を含んではならない．

(機能 2)　天気予報の情報を返信する機能

この機能は，ユーザが “@masabot (日にち) の (都道府県名) の天気” と投稿したとき，指定した “(日にち)” における “(都道府県名)” の天気予報の情報を返信する．例として，ユーザが “@masabot 明日の東京都の天気” と投稿したときの，SlackBot の返信を図 3 に示す．この機能を使用するにあたって，ユーザが指定できる “(日にち)” と “(都道府県名)” を表 1 に示す．ユーザは，表 1 から “(日にち)” に入れる文字列を 1 つ指定する．また，“(都道府県名)” には，47 都道府県から都道府県名を 1 つ指定する．これらの情報を用いて，ユーザは Slack に “@masabot (日にち) の (都道府県名) の天気” と投稿する．これにより，SlackBot は指定された “(日にち)” における “(都道府県名)” の天気予報の情報を返信する．このときに返信する天気予報の情報は，Web サービスである Weather Hacks（気象データ配信サービス）[2] を利用して取得している．

ユーザが (機能 1)，(機能 2) のどちらにも当てはまらない文章を Slack に投稿したとき，本プログラムは以下の文章を返信する．

こんにちは、masabot です。
"@masabot 「○○○」と言って"と投稿した場合、"○○○"と返信します。
天気予報を知りたいときは、"@masabot □□の△△の天気"
(□□は「今日」「明日」「明後日」のいずれか)(△△は都道府県名) と投稿してください。
この日の指定した都道府県の天気を返信します。

表 2　動作環境

項目	内容
OS	Ubuntu 18.04.4 LTS
CPU	Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz
メモリ	512MB
Ruby	ruby 2.6.6p146
Gem	activesupport 6.0.2.2
	bundler 2.0.2
	mustermann 1.1.1
	rack 2.2.2
	rack-protection 2.0.8.1
	ruby2_keywords 0.0.2
	sinatra 2.0.8.1
	tilt 2.0.10

6　動作環境

本プログラムは Heroku 上で動作する．Heroku とは，プログラムをデプロイすることにより，クラウド上でプログラムを実行することができるプラットフォームである．Heroku の動作環境を表 2 に示す．

7　環境構築

7.1　概要

本プログラムを動作させるための環境の構築手順を以下に記述する．

(1) Slack の Incoming WebHook の設定
(2) Slack の Outgoing WebHook の設定
(3) Heroku アカウントの作成と設定

上記の構築手順の詳細を以下に記述する．

7.2　Slack の Incoming WebHook の設定

Slack における Incoming WebHook の設定を以下の手順で行う．

(1) 自分の Slack アカウントにログインする.
(2) Slack の左上のメニューを開き,「その他管理項目」の「アプリを管理する」を開く.
(3) カスタムインテグレーションから,「Incoming WebHook」を開く.
(4)「Slack に追加」を選択した後, Incoming WebHook がメッセージを投稿するチャンネルを選択し, インテグレーションを追加する.
(5)「名前をカスタマイズ」にて, 設定する Incoming WebHook の名前を入力する. また, WebHook URL を 7.4 節 (9) で使用するので控えておく.
(6)「設定を保存する」を選択する.

7.3 Slack の Outgoing WebHook の設定

Slack における Outgoing WebHook の設定を以下の手順で行う.

(1) 自分の Slack アカウントにログインする.
(2) Slack の左上のメニューを開き,「その他管理項目」から「アプリを管理する」を開く.
(3) カスタムインテグレーションから,「Outgoing WebHook」を開く.
(4)「インテグレーションを追加する」を選択する.
(5) Outgoing WebHook に関して, 以下の設定を行う.
   (A)「チャンネル」にて, ユーザが投稿した特定の文字列を監視するチャンネルを設定する.
   (B)「引き金となる言葉」にて, WebHook が動作する契機となる文字列を入力する.
   (C)「名前をカスタマイズ」にて, 設定する Outgoing WebHook の名前を入力する.
(6)「設定を保存する」を選択する.

7.4 Heroku アカウントの作成と設定

Heroku アカウントの作成と設定を以下の手順で行う.

(1) 以下の URL より Heroku にアクセスし,「新規登録」より Heroku アカウントを登録する.

```
https://jp.heroku.com/
```

(2) 登録したアカウントでログインし,「Getting Started on Heroku」にて,「Ruby」を選択する.
(3) 以下のコマンドを実行し, Heroku CLI をインストールする.

```
$ sudo snap install heroku --classic
```

(4) `heroku` コマンドを使用するために, 以下のコマンドを実行し, パスを通す.

```
$ export PATH=$PATH:/snap/bin/
```

(5) 以下のコマンドを実行し, Heroku CLI がインストールできていることを確認する.

```
$ heroku version
heroku/7.42.0 linux-x64 node-v12.16.2
```

上記の例では，heroku/7.42.0 linux-x64 node-v12.16.2 の Heroku CLI がインストールできていることを示している．

(6) 以下のコマンドを実行し，Heroku CLI にログインする．

```
$ heroku login
```

(7) 作成したアプリケーションのディレクトリに移動して，以下のコマンドを実行し，Heroku 上にアプリケーションを生成する．

```
$ heroku create <myapp_name>
```

<myapp_name>にはアプリケーション名を入力する．なお，アプリケーション名は小文字，数字，およびハイフンのみ使用できる．

(8) 以下のコマンドを実行し，生成した Heroku のアプリケーションが，登録されていることを確認する．

```
$ git remote -v
heroku https://git.heroku.com/<myapp_name>.git (fetch)
heroku https://git.heroku.com/<myapp_name>.git (push)
```

(9) 以下のコマンドを実行し，7.2 節 (5) で控えておいた Incoming WebHook URL を，Heroku の環境変数に追加する．

```
$ heroku config:set INCOMING_WEBHOOK_URL="https://XXXXXXXXXXXX"
```

上記のコマンドにおける"https://XXXXXXXXXXXX"に，7.2 節 (5) で控えておいた Incoming WebHook URL を入力する．

8 使用方法

本プログラムを使用するには，以下の手順で行う．

(1) 以下に示す Git リポジトリから本プログラムを取得する．

```
git@github.com/masafumi0612/BootCamp.git
```

本プログラムは，上記の Git リポジトリ内の 2020/SlackBot/SlackBot.rb 及び，2020/SlackBot/MySlackBot.rb である．

(2) 以下のコマンドを用いて，本プログラムを Heroku にデプロイする．

```
$ git push heroku master
```

9 エラー処理

本プログラムにおけるエラー処理を以下に記述する．

(1) (機能 2) を使用する際に，ユーザが投稿する文字列の中に (日にち) が含まれていないときや，使用できる “今日”，“明日” および “明後日” 以外の文字列を使用したときは，“今日” の指定した都道府県の天気予報を返信する．例えば，“東京都の天気” や “明々後日の東京都の天気” という文字列を投稿した際は，今日の東京都の天気予報を返信する．

(2) (機能 2) を使用する際に，ユーザが投稿する文字列の中に都道府県名が含まれていないとき，本プログラムは以下のように返信する．

> **都道府県名を入力してください。**
> **天気予報を知りたいときは、"@masabot □□の△△の天気"**
> **（□□は「今日」「明日」「明後日」のいずれか）（△△は都道府県名） と投稿してください。**
> **指定した日にちの都道府県の天気を返信します。**

これに当てはまる例として，ユーザが “@masabot 明日の天気” という文字列を投稿したときが挙げられる．

10 保証しない動作

本プログラムの保証しない動作を以下に記述する．

(1) 6 章に示した動作環境以外の環境で，本プログラムを実行したときの動作

(2) 本プログラムが，Slack の Outgoing WebHook 以外から POST リクエストを受け取って実行した際の動作

参考文献

[1] Slack: その仕事、Slack で。，Slack Technologies.Inc（オンライン），入手先 ⟨https://slack.com/intl/ja-jp/⟩（参照 2020-05-15）．

[2] Hacks, W.: Weather Hacks（気象データ配信サービス），日本気象協会（オンライン），入手先 ⟨http://weather.livedoor.com/weather_hacks/⟩（参照 2020-05-15）．