

# SAMIT18.05

GAS による初めてのボット作成

17043052 藤原 溪亮

2018 年 5 月 26 日

# 今日やること

ボット作ります

# 今日やること

こんな感じです.

The screenshot displays a Slack interface. On the left is a dark sidebar with the following elements: a header 'SAMIT18.05Bot' with a dropdown arrow and a bell icon; a list of channels including 'FujiwaLaTeX', '全スレッド', 'チャンネル' (with a plus icon), '# general' (highlighted in green), '# random', 'ダイレクトメッセージ' (with a plus icon), 'slackbot', 'FujiwaLaTeX (自分)', '+ メンバーを招待する', and 'App' (with a plus icon). The main area shows the '#general' channel header with a star icon, a count of 1 member, and a description '全社的なアナウンスと業務関連の事項'. Below the header is a blue banner for '5月25日の18:02以降、3件の新しいメッセージがありました' with a '既読にする' button. The message list starts with a thumbs-up icon and the text '赤城みりあ アプリ 16:52', followed by a repeating pattern of '基礎物理A', '情報リテラシー演習', and '基礎物理A'.

SAMIT18.05Bot

FujiwaLaTeX

全スレッド

チャンネル

# general

# random

ダイレクトメッセージ

slackbot

FujiwaLaTeX (自分)

+ メンバーを招待する

App

#general

☆ | 1 | 0 | 全社的なアナウンスと業務関連の事項

今日

5月25日の18:02以降、3件の新しいメッセージがありました 既読にする ×

赤城みりあ アプリ 16:52

基礎物理A

基礎物理A

情報リテラシー演習

情報リテラシー演習

基礎物理A

基礎物理A

情報リテラシー演習

情報リテラシー演習

基礎物理A

基礎物理A

情報リテラシー演習

情報リテラシー演習

基礎物理A

基礎物理A

情報リテラシー演習

情報リテラシー演習

基礎物理A

基礎物理A

情報リテラシー演習

情報リテラシー演習

基礎物理A

基礎物理A

# 今日やること

Slack(チャットツール)に定期的に  
時間割を通知してくれるボットを  
作ります.

# Slack の用意

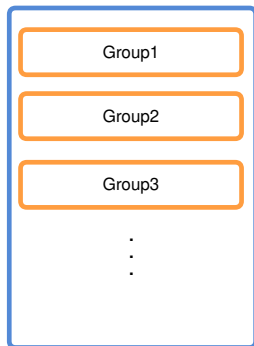
## Slack とは

Slack Technologies 社が開発したビジネス向け  
チャットツール

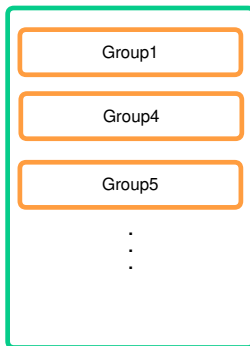


## LINE

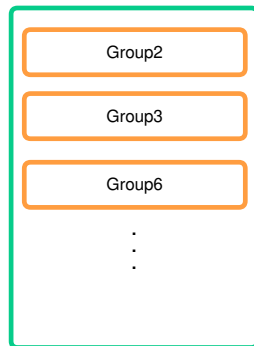
myAccount



Account1

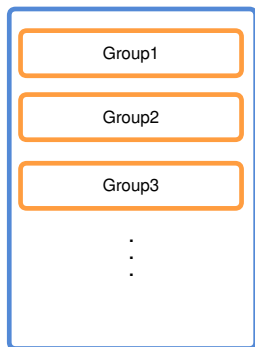


Account2

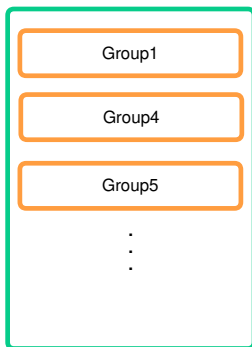


## LINE

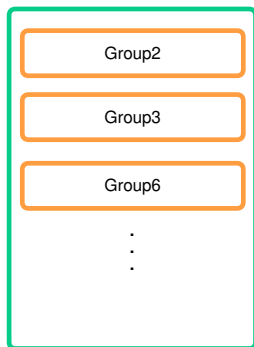
myAccount



Account1



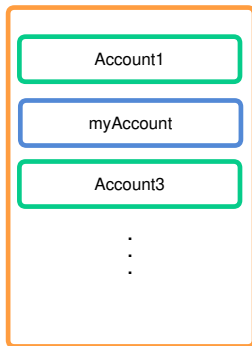
Account2



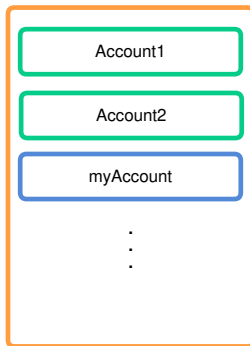
一つのアカウントが  
複数のグループに所属

## Slack

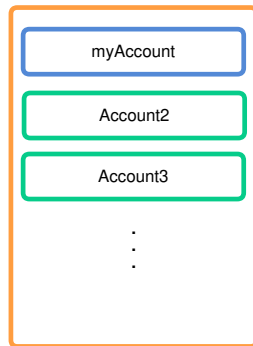
### WorkSpace\_A



### WorkSpace\_B



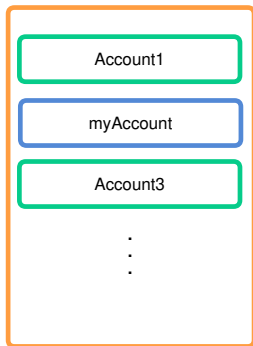
### WorkSpace\_C



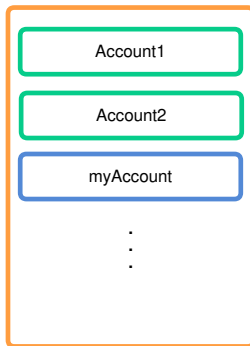


## Slack

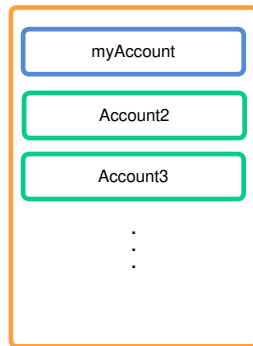
WorkSpace\_A



WorkSpace\_B



WorkSpace\_C



# グループ毎にアカウントを持つ

# ワークスペース作成

- ① `https://slack.com/intl/ja-jp` にアクセス
- ② 右上の「ワークスペース」をクリック
- ③ 最下部にある「ワークスペースの作成」をクリック

# アプリ追加

- ① チャット画面から「+アプリ追加」をクリック
- ② “incomming web hook” で検索
- ③ 「設定を追加」をクリック

## 着信 Web フック

外部から Slack にメッセージを送れるアプリ

# アプリ追加

- ① チャット画面から「+アプリ追加」をクリック
- ② “incomming web hook” で検索
- ③ 「設定を追加」をクリック

## 着信 Web フック

外部から Slack にメッセージを送れるアプリ

投稿するチャンネルを決めたら「インテグレーションの追加」

# Slack にメッセージを送ってみる

## コマンドラインからメッセージを送信

```
1 curl -X POST --data-urlencode  
2 "payload={\"text\": \"Hello Slack\"}"  
3 <your Webhook URL>
```

# Slack にメッセージを送ってみる

## コマンドラインからメッセージを送信

```
1 curl -X POST --data-urlencode  
2 "payload={\"text\": \"Hello Slack\"}"  
3 <your Webhook URL>
```

Windows は curl コマンドありません。

# Google スプレッドシートをオリジナル 時間割にする

## スプレッドシートをコピー

- ① hogehoge.com からスプレッドシートにアクセス
- ② 「ファイル」 → 「コピーを作成」
- ③ 自分のディレクトリに保存
- ④ コピー先で時間割カスタマイズ

エクセルと使い方はほぼ一緒

# GAS について

## GAS とは

Google Apps Script の略で，Google のサービス进行操作できる環境．言語は JavaScript

## GAS から利用できるサービス

Google スプレッドシート，Google ドキュメント，Google スライドのデータ操作や取得，Gmail へ通知を送信などが可能



# スプレッドシートに GAS を導入

- ① コピーしたスプレッドシートを開く
- ② 「ツール」 → 「スクリプトエディタ」

# GAS を書く

Hello World!!

```
1 function helloWorld(){  
2   Logger.log("Hello World!!");  
3 }
```

## スクリプト実行

- ① 「実行」 → 「関数を実行」 → “myFunction”
- ② ctrl + Enter でログ確認

# GAS を書く

## スプレッドシートからデータ取得

```
1 function helloWorld(){
2   Logger.log("Hello World!!");
3 }
4 function getSheetInfos(){
5   var values = SpreadsheetApp.getActiveSheet().getRange("A1").getValues();
6   return values;
7 }
```

A1 セルから値を取得して，ログに出力  
(ctrl+Enterで見れる)

# GAS から Slack へメッセージを送る

## GAS から POST リクエストを送る

```
1 function postSlack(text){  
2   var url = "<your Webhook URL>";  
3   var options = {  
4     "method" : "POST",  
5     "headers": {"Content-type": "application/json"},  
6     "payload" : JSON.stringify({"text": text})  
7   };  
8   UrlFetchApp.fetch(url, options);  
9 }
```

# GAS から Slack へメッセージを送る

## GAS から POST リクエストを送る

```
1 function postSlack(text){  
2   var url = "<your Webhook URL>";  
3   var options = {  
4     "method" : "POST",  
5     "headers": {"Content-type": "application/json"},  
6     "payload" : JSON.stringify({"text": text})  
7   };  
8   UrlFetchApp.fetch(url, options);  
9 }
```

## 送るメッセージを受け取る関数

```
1 function test(){  
2   postSlack("これはテストです");  
3 }
```

# GAS から Slack へメッセージを送る

## GAS から POST リクエストを送る

```
1 function postSlack(text){  
2   var url = "<your Webhook URL>";  
3   var options = {  
4     "method" : "POST",  
5     "headers": {"Content-type": "application/json"},  
6     "payload" : JSON.stringify({"text": text})  
7   };  
8   UrlFetchApp.fetch(url, options);  
9 }
```

## UrlFetchApp.fetch

url で指定した場所に options で指定した条件で  
データを送る

# GAS から Slack へメッセージを送る

## GAS から POST リクエストを送る

```
1 function postSlack(text){  
2   var url = "<your Webhook URL>";  
3   var options = {  
4     "method" : "POST",  
5     "headers": {"Content-type": "application/json"},  
6     "payload" : JSON.stringify({"text": text})  
7   };  
8   UrlFetchApp.fetch(url, options);  
9 }
```

url

データを送る先

# GAS から Slack へメッセージを送る

## GAS から POST リクエストを送る

```
1 function postSlack(text){  
2   var url = "<your Webhook URL>";  
3   var options = {  
4     "method" : "POST",  
5     "headers": {"Content-type": "application/json"},  
6     "payload" : JSON.stringify({"text": text})  
7   };  
8   UrlFetchApp.fetch(url, options);  
9 }
```

### options

**method** HTTP リクエストの種類

**headers** 送るデータの形式

**payload** 送るデータの内容



# GAS から Slack へメッセージを送る

## GAS から POST リクエストを送る

```
1 function postSlack(text){  
2   var url = "<your Webhook URL>";  
3   var options = {  
4     "method" : "POST",  
5     "headers": {"Content-type": "application/json"},  
6     "payload" : JSON.stringify({"text": text})  
7   };  
8   UrlFetchApp.fetch(url, options);  
9 }
```

## JSON

データ記述言語の一種 (データ構造を表す)

# 月曜日の時間割を Slack に通知してみる

月曜日の時間割 → B2 から B11

## 月曜日の時間割を返す関数

```
1 function notifyMondaySchdule(){  
2   var classes = getSheetInfos("B2:B11");  
3   var todayScheduleStr = listToStr(classes);  
4   postSlack(todayScheduleStr);  
5 }
```

## リストを文字列にする関数

```
1 function listToStr(strList){  
2   const str = strList.reduce(  
3     function(previousValue, currentValue){  
4       return previousValue + "\n" + currentValue;  
5     }  
6   );  
7   return str;  
8 }
```

# 今日の時間割を Slack に通知してみる

## 今日の時間割を返す関数

```
1 function notifyTodaySchedule(){
2   const weekDayList = {
3     '日': 'B', '月': 'B', '火': 'C', '水': 'D', '木': 'E', '金': 'F', '土': 'B'
4   };
5   const today = getDay();
6   const rowNumber = weekDayList[today];
7   const classes = getSheetInfos(rowNumber+"2:"+rowNumber+"11");
8   const todayScheduleStr = listToStr(classes);
9   postSlack(todayScheduleStr);
10 }
```

## 今日の曜日を返す関数

```
1 function getDay(){
2   const weekDayList = ["日", "月", "火", "水", "木", "金", "土"];
3   const dateObj = new Date() ;
4   const weekDay = weekDayList[ dateObj.getDay() ];
5   return weekDay;
6 }
```

# 定期的に時間割を Slack に通知してみる

- ① 時計のアイコンをクリック → 「トリガーが設定...」をクリック
- ② 実行 → “notifyTodaySchedule” へ
- ③ イベント → 「時間主導型」, 「分タイマー」, 「1 分ごと」
- ④ 保存

約 1 分待つ

# 携帯に通知を送る

- ① Android/iPhone に “Slack” をインストール
- ② “Slack” を開いて「ワークスペースを追加」
- ③ 今回作成したワークスペースに登録したアドレスを入力
- ④ 「参加しているワークスペース」からワークスペース選択