

# 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 for 'SAMIT18.05Bot' with a dropdown arrow and a bell icon; a green dot next to 'FujiwaLaTeX'; a search bar containing '全スレッド'; a 'チャンネル' (Channels) section with a plus icon, listing '# general' (highlighted in green) and '# random'; a 'ダイレクトメッセージ' (Direct Messages) section with a plus icon, listing 'slackbot' and 'FujiwaLaTeX (自分)'; a '+ メンバーを招待する' (Invite members) button; and an 'App' section with a plus icon.

The main area shows the '#general' channel. At the top, it says '☆ | 👤 1 | 📄 0 | 全社的なアナウンスと業務関連の事項' and '今日'. Below this is a blue notification bar: '↓ ジャンプ 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 ×'. The message list contains several entries, each starting with a thumbs-up emoji and the text '赤城みりあ アプリ 16:52', followed by '基礎物理A'. The messages are grouped into three sets of three, with the first set having a '情報リテラシー演習' message above the '基礎物理A' messages.

# 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

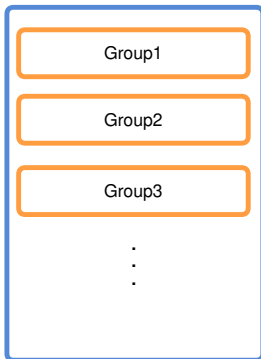
## Slack とは

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

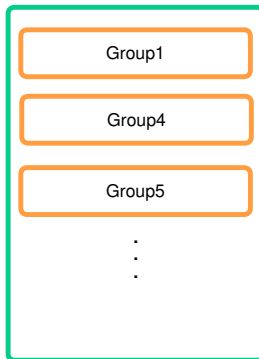


## LINE

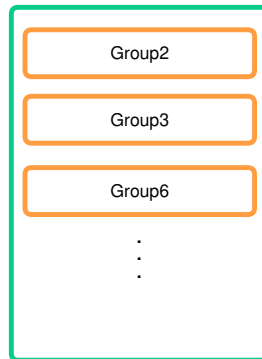
myAccount

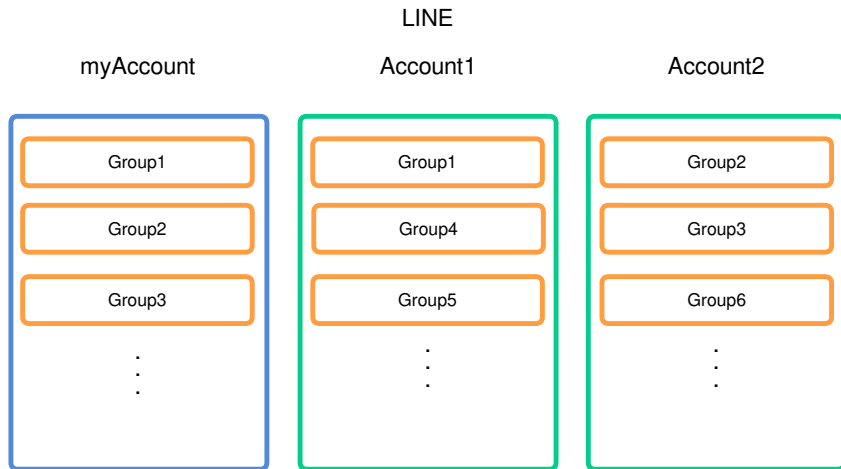


Account1



Account2

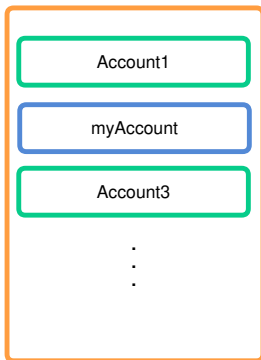




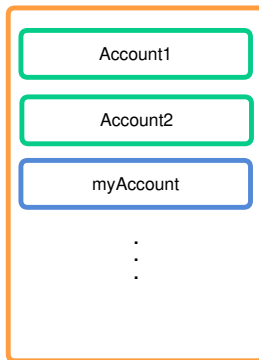
一つのアカウトが複数のグループに所属

## Slack

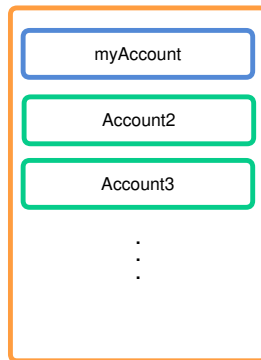
### WorkSpace\_A



### WorkSpace\_B



### WorkSpace\_C



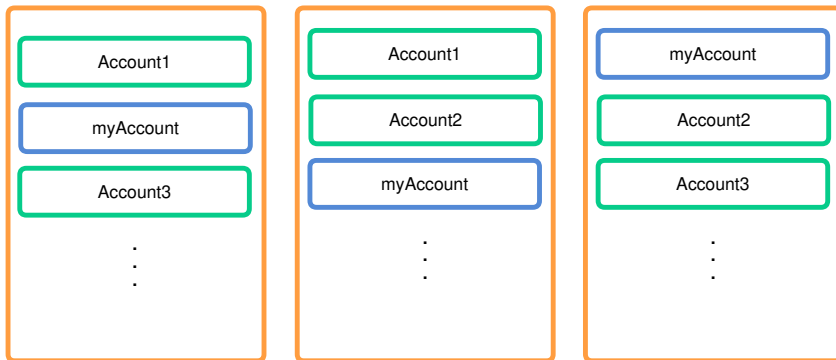


## Slack

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 -d  
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 -d  
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 <https://goo.gl/ec1Gmk> 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」 → “helloworld”
- 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   const 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   const url = "<your Webhook URL>";  
3   const 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   const url = "<your Webhook URL>";  
3   const 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   const url = "<your Webhook URL>";  
3   const 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   const url = "<your Webhook URL>";  
3   const 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   const url = "<your Webhook URL>";  
3   const 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   const url = "<your Webhook URL>";  
3   const 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   const classes = getSheetInfos("B2:B11");
3   const 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 strs = "1" + strList.reduce(
3     function(previousValue, currentValue, index){
4       return previousValue + "\n" + String(index+1) + currentValue;
5     });
6   return strs;
7 }
```

# 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": "B2:B11", "月": "B2:B11", "火": "C2:C11", "水": "D2:D11", "木": "E2:E11", "金": "F2:F11", "土": "B2:B11",
4   };
5   const today = getDay();
6   const range = weekDayList[today];
7   const classes = getSheetInfos(range);
8   postSlack(listToStr(classes));
9 }
```

## 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