

# Gmail APIでメールを扱おうとしたら結構辛かった話

ITネタ・自動化ネタ ライトニングトーク～アマギフ1万円争奪戦～

RPACommunity YouTube登録者数1万名達成記念 特別イベント！

2024-02-09

@hrs\_sano645

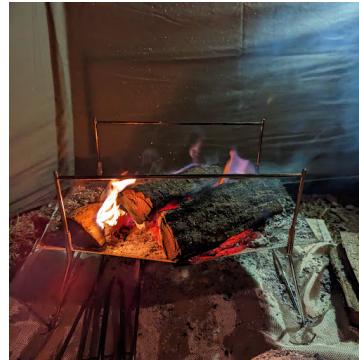
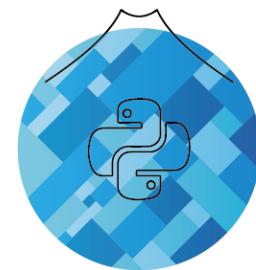
# お前誰よ / Self Introduction

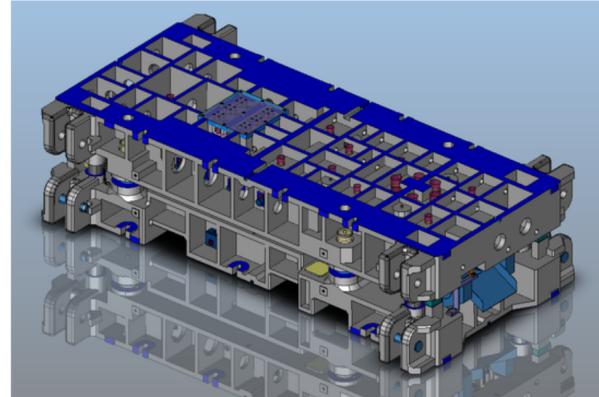
佐野浩士 (Hiroshi Sano) [@hrs\\_sano645](#)

- 🗺️: 静岡県富士市🗻
- 🏢: 株式会社佐野設計事務所 代表取締役
- 💬🤝
  - 🐍: PyCon mini Shizuoka Stuff / Shizuoka.py / Unagi.py / Python駿河
  - CivicTech, Startup Weekend Organizer
- Hobby: Camp🏕, DIY🛠, IoT💡



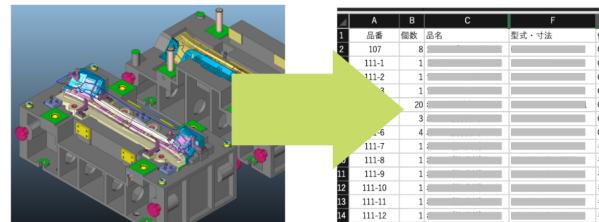
**PyCon  
mini  
SHIZUOKA**





## 3D機械設計 / 3Dモデリング

自動車向けプレス金型の3D設計  
金型設計関連の3Dモデリング  
製品データの3Dモデリング



## 製造業DX支援



設計データをデジタル化  
製造業DX取り組みサポート

なんで会社の紹介をしたかというと

**2023年の社内個人的目標: 業務効率化を限界まで進める**

# 効率化する理由

- とある依頼ベースの案件業務
- 今までではそれほど多くなかったが、**年を跨いで急激に増える**
  - 人力でやっていては**追いつかん**そう。やばい
- 人間が必要な部分以外、**あらゆる手作業を止める！** → 成功した!



# どんなことを効率化？

- **手作業していたこと自動化**

- 依頼受注（メール）→ボイラープレートツールで作業プロジェクトフォルダーを生成
- Googleスプレッドシート連携して案件管理
- 会計サービスと連携して見積書/請求書生成（書類作成）
- 依頼企業側のシステム連携: WEBスクレイピング

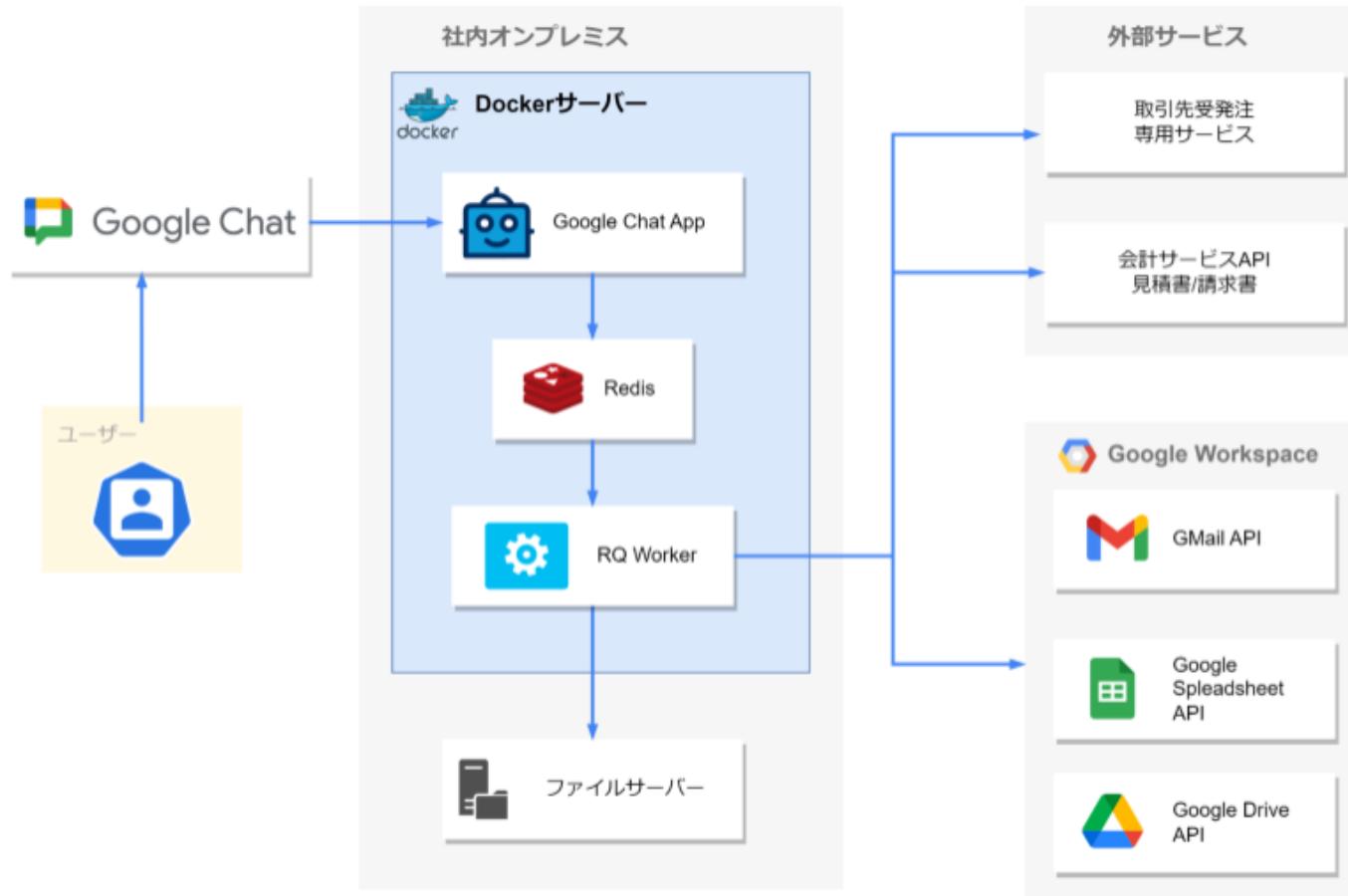
- **自動処理の実行をChatOpsで**

- 処理を動かしていくか確認した上で実行させる
- Google Chatでチャットボット作成
- スマホから実行もできる

究極には指一本で仕事をしたい



# 業務タスク自動化サービスの構成



自動化の最初の一歩としてたくさん届くメールを扱う

メールを人力で扱う問題点

- メールの内容を見てアクション起こす
- 見落としたりする
- 必要な情報を取りこぼす（添付ファイルとか）

人で頑張りたくない→プログラムで自動化しよう！

もちろんRPAやノーコードでも！

Gmailを使っているのでGmail APIを扱ってみよう！

Gmail APIとは

Gmailをプログラムから操作するAPI



APIの使ってメールの内容を収集する流れ

REST APIで各プログラミング言語のSDKでやります。

- メールのリストを取得して: `user.message.list` メソッド
- メールのメッセージを取得する: `user.message.get` メソッド

ここまであなんとかなります。

問題はここから

取得した先のメールメッセージのデータ構造

結構メール（RFC定義）のデータ構造そのままっぽい

mimetypeが立ちはだかる

`mimetype`とは、データの種類を表す文字列

メールは、テキスト、HTML、画像、添付ファイルなどが混在している。

メールのデータは構造化されていて、それぞれmimetypeで表現、判断ができる

参考: <https://www.softel.co.jp/blogs/tech/archives/5726>

## mimetypeの種類

- text/plain
- text/html
- image/\*\*
- application/\*\*
- multipart/\*\* (これが厄介)

テキストメール、htmlメール、添付ファイル、画像入ってる、などなど

multipartの種類で入れ子構造が変わる

- multipart/alternative: メールにあるテキストとHTML
- multipart/mixed: メールと添付ファイル
- multipart/related: HTMLと画像

multipartでも組み合わせで入れ子構造になっている

そして、Gmail API側の構造を見ると…

```
1  {
2    "id": "1866d9ce947765f4",
3    "threadId": "1866d9ce947765f4",
4    "labelIds": ["IMPORTANT", "Label_14", "CATEGORY_PERSONAL"],
5    "snippet": "*****",
6    "payload": {
7      "partId": "", "mimeType": "multipart/mixed", "filename": "",
8      "headers": [...]
233    ],
234    "body": {
235      "size": 0
236    },
237    "parts": [
238      {
239        "partId": "0", "mimeType": "multipart/alternative", "filename": "", "headers": [...]
240      ],
241      "body": {...}
242    },
243    "parts": [
244      {"partId": "0.0", "mimeType": "text/plain", "filename": "", "headers": [...]
245      },
246      "body": {...}
247    ],
248    "parts": [
249      {"partId": "0.1", "mimeType": "text/html", "filename": "", "headers": [...]
250      },
251      "body": {...}
252    ],
253    "body": {
254      "size": 17966,
255      "data": "PGh0bWwgeG1sbmM6dj0idXJuOnNjaGVtYXMtbWljcm9zb2Z0LWNvbTp2bWwiIHhtbG5zOm89InVybjpzY2hlbWFzLW1pY3:
```

入れ子構造どんだけ... 😱

(この辺、RPAなアプリやサービスは抽象化がされていると思うので、ここまでデータ構造を意識することはないかも)

試しに届いてるメールを100件ほど取得して、  
どんな入れ子構造のパターンがあるのかみてみる

結果

multipart/alternative

  text/plain

  text/html

  : 62

text/plain

  : 22

text/html

  : 5

multipart/mixed

  multipart/alternative

    text/plain

    text/html

  : 4

multipart/signed

  multipart/alternative

    text/plain

    text/html

  application/x-pkcs7-signature

  : 2

これ全部判別対応するの？メール怖いよお😭



それでも頑張らないといけないので



最後に欲しい情報を狙い撃ちするPythonのコード

メールの中にあるだろう、テキストでの本文を取得する

```
def find_message_parts_text(message, message_parts=None):
    """
    メッセージから text/plain と text/html の部分を再帰的に探索する関数
    """
    if message_parts is None:
        message_parts = {"text/plain": None, "text/html": None}

    mimetype = message.get("mimeType")
    data = message.get("body", {}).get("data")

    if mimetype == "text/plain" and data:
        message_parts["text/plain"] = base64.urlsafe_b64decode(data).decode("utf-8")
    elif mimetype == "text/html" and data:
        message_parts["text/html"] = base64.urlsafe_b64decode(data).decode("utf-8")
    #
    for part in message.get("parts", []):
        find_message_parts_text(part, message_parts)

    return message_parts
```

```
# Gmail APIのPythonクライアントでメッセージのIDをもとにメールを取得
# `user.message.list`でメールのIDを取得し`user.message.get`を使って本文を取得
message = (
    service.users()
    .messages()
    .get(userId="me", id=[APIで手に入れたメールのID], format="full")
    .execute()
)

# messageのpayloadからmimetypeを元に、本文を取得
msg_payload = message["payload"]

# 探索的にtext/plainを探して表示する。text/htmlのみのメールもあるので注意
message_parts = find_message_parts_text(msg_payload)
message_text = message_parts["text/plain"] or message_parts["text/html"]
if message_text:
    # 20文字まで出している
    print(f'{message_text[0:20]}\n')
```

詳しくはブログにまとめたのでこちらからどうぞ！

<https://hr-sano.net/blog/gmail-api-intro/>

まとめ



メールのデータ構造を理解すること  
でも色々あるから怖い。辛い。  
欲しいものを探索して狙い撃ちするといいかも