

ご当地グルメマップを作ろう

Lets create Local Food Map

PyCon APAC 2023 Day2

Hiroshi Sano / 佐野浩士



Agenda

- トークのモチベーション
- 情報を集めよう: どの情報を使うか
- マップデータを作ろう: 情報収集をアウトプット
- 利用先を考えよう: サービスと連携しよう

Today's Document

付録含めてすでに公開されています。

- GitHub: Starくれー！

<https://github.com/hrsano645/pyconapac2023-local-food-map>

- スライド

<https://github.com/hrsano645/pyconapac2023-local-food-map/slides.pdf>

あとでトライしたい方は参考にしてみてください。

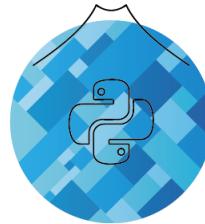
Self Infroduction

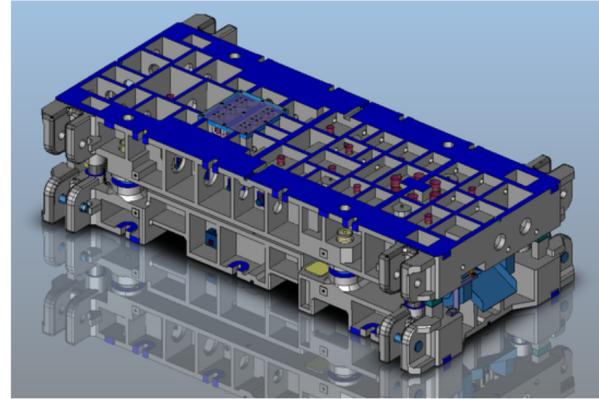
Hiroshi Sano(佐野浩士) [@hrs_sano645](#)

- 🏢: 株式会社佐野設計事務所 CEO
- 🐍: PyCon mini Shizuoka Stuff
 - Shizuoka.py / Unagi.py / PythonSuruga
- CivicTech, [Startup Weekend](#) Organizatior
- Hobby: Camp, DIY, IoT 



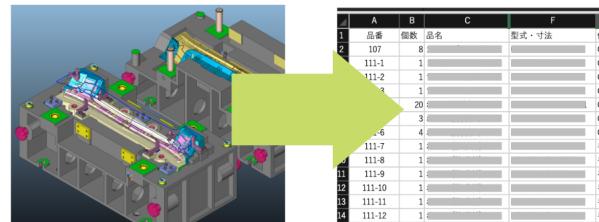
PyCon
mini
SHIZUOKA





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

トークのモチベーション

- ご当地のグルメの**情報収集してマップを作りましょう！食べに行きましょう！**
- PyCampを終えた人向け: Pythonでデータを集めて作り利用するプロセス
を学べます

PyCamp:Python Boot Campとは

- ・日本全国で開催されているPythonの学習プログラム
- ・半日をかけて、Pythonの基礎を学び、簡単なプログラムを作る
 - 地元静岡県だと3回開催されている



このトークはその続きからトライできるコンテンツを目指して作りました

ご当地グルメ？

- 旅行がしやすくなった昨今
- 観光地でご飯食べようにも色々悩む
 - 情報サイトは有名なものだけ。美味しいところは他にもある
- 公開されている情報をもとにマップを作って旅行計画するのはいかがでしょう？

いわゆる**B級グルメ**を食べましょう 😊

今回のお題

富士宮焼きそば



- 主に静岡の富士宮市周辺で食べられる焼きそば
- B級グルメグランプリは殿堂入り
- 麺は富士宮周辺でしか手に入らないのでまさにローカルフード
 - (通販もあるけどね)
- 最近NYで開催されたコナモングランプリで優勝した

家庭料理としても食べられる



地元の人は、多分月に数回は食べてる

ところで、みなさん

もう食べたくなったでしょ？ 😭

このあとはパーティ🎉ですし、お腹空きましたね 😭

富士宮焼きそばマップを作りましょう！

今回のトークで目指すこと

- お店情報を探してWEBスクレイピング
- データを整形 -> CSVファイルにする
- Googleマイマップで呼び出す

データを読む/取り込む



データを加工/出力する



データを使う/活用する

今回のトークで目指すこと

- お店情報を探してWEBスクレイピング
- データを整形 -> CSVファイルにする
- Googleマイマップで呼び出す

↓

- データを読む/取り込む
- データを加工/出力する
- データを使う/活用する

この流れで説明していきます。

ご当地グルメの情報はどこにあるか

- 地域の情報を収集
- その情報は機械可読性があるか

観光情報を見てみる

- 市役所
- 観光協会
- ご当地グルメの公式サイト（よく〇〇学会とも言われる）

グルメ情報サイトを見たらそこで終了ですよ

富士宮焼きそば学会 公式サイトを例にします。

<https://umya-yakisoba.com/>

データを読む/取り込む



データを加工/出力する



データを使う/活用する

データを読む/取得する

- WEBスクレイピングで収集する
- [付録]画像識別で加工を試みる

機械可読性はどちらも微妙

WEBスクレイピングはまだやりやすい

WEBスクレイピングの注意点

※ただし、多数のアクセスはしないように注意

- 単純に迷惑かけがち
- トラブルになりがち
- 試すときは少し時間を置きながらアクセスしましょう
 - ランダム時間置いてみるとか
 - 回数リミットをつけて待つようとする
 - サイトポリシーがあれば従う

利用するライブラリ

- requests: HTTPアクセス→情報取得（今回はHTML）
- BeautifulSoup4: HTML（マークアップ言語）解析と抽出

```
pip install requests  
pip install beautifulsoup4
```

*:スライドのコードは説明向けです。そのままだと動かないこともあります
資料のリポジトリから動作するスクリプトをDL可能です。

サイトの構造を見てみましょう

この構造からbeautifulsoup4を使って必要な情報を取り出します。



構造の中にあるタグから必要な情報を取得する

```
import requests
from bs4 import BeautifulSoup

shopinfo_list = []
res = requests.get(url)
soup = BeautifulSoup(res.text, 'html.parser')

# ここではdiv.p-shopList > a にURLとその中にお店情報がまとまっているので、aタグから取り出す
shopinfo_tags = soup.find('div', class_='p-shopList').find_all("a")
```

aタグの下にあるそれぞれのタグから情報取得

```
for shopinfo_tag in shopinfo_tags:  
    shopdata = {}  
    # aタグの子要素となるdivは上から店名、住所、電話番号、定休日。  
    # ここではurlと店名だけまとめたリストを作る  
    shopdata['specurl'] = shopinfo_tag.get('href')  
    shopdata['店名'] = shopinfo_tag.find_all("div")[1].text  
    shopinfo_list.append(shopdata)
```

取得結果

```
>>> shopinfo_list
[{'specurl': 'https://umya-yakisoba.com/shop/3776/', '店名': 'お好焼\u3000あき'},
 {'specurl': 'https://umya-yakisoba.com/shop/3777/', '店名': 'あさ家'},
 {'specurl': 'https://umya-yakisoba.com/shop/3774/', '店名': 'あるばとろす'},
 {'specurl': 'https://umya-yakisoba.com/shop/3616/', '店名': 'いっぷく亭'},
 {'specurl': 'https://umya-yakisoba.com/shop/3772/', '店名': 'お好み食堂 伊東'},
 ...
]
```

店名にある空白などを取り除きます

```
def replace_str(text: str) -> str:  
  
    replace_str_map = {  
        # 置き換える  
        "\u3000": " ",  
    }  
    replaced_text = text  
    for key, val in replace_str_map.items():  
        replaced_text = replaced_text.replace(key, val)  
    return replaced_text
```



```
for shopinfo_tag in shopinfo_tags:  
    shopdata = {}  
    # divは上から店名、住所、電話番号、定休日。  
    # ここではurlと店名だけまとめたリストを作る  
    shopdata['specurl'] = shopinfo_tag.get('href')  
    shopdata['店名'] = replace_str(shopinfo_tag.find_all("div")[1].text)  
  
    shopinfo_list.append(shopdata)
```

↓

```
>>> shopinfo_list  
[{'specurl': 'https://umya-yakisoba.com/shop/3776/', '店名': 'お好焼 あき'},  
 {'specurl': 'https://umya-yakisoba.com/shop/3777/', '店名': 'あさ家'},  
 {'specurl': 'https://umya-yakisoba.com/shop/3774/', '店名': 'あるばとろす'},  
 {'specurl': 'https://umya-yakisoba.com/shop/3616/', '店名': 'いっぷく亭'},  
 ...]
```

収集した詳細URLのリストを使って、お店情報収集

ホーム > 富士宮やきそば店一覧 > お好焼 あき



```
要素 コンソール ソース ネットワーク
▼<div id="content" class="content cf">
  ▼<div id="content-in" class="content-in wrap">
    ▼<main id="main" class="main" itemscope itemtype="http://schema.org/LocalBusiness">
      ▶<div id="breadcrumb" class="breadcrumb breadcrumb_main">
        ...</div>
      <!-- /#breadcrumb -->
      ▼<article>
        ▶<header class="article-header entry-header">
          <div class="p-shopPh"> </div> flex
        ▼<dl class="p-shopDetails"> flex
          ...
          <dt>エリア</dt> == $0
          ▶<dd>...</dd>
          <dt>お店名ふりがな</dt>
          <dd> あき </dd>
          <dt>地図</dt>
          <dd> B6 </dd>
          <dt>業種</dt>
          <dd> 飲食店 </dd>
          <dt>住所</dt>
          <dd> 富士宮市野中東町112-1 </dd>
          <dt>TEL</dt>
          <dd> 0544-27-0004 </dd>
          <dt>営業時間</dt>
          ▶<dd>...</dd>
          <dt>定休日</dt>
          <dd> 火曜日 </dd>
          <dt>受入人数</dt>
          <dd> 12 </dd>
          <dt>駐車場</dt>
          <dd> 4 </dd>
          <dt>焼き方</dt>
          <dd> お店 </dd>
          <dt>料金目安</dt>
          <dd> 350~600円 </dd>
          <dt>調査員:よしむらめぐみ</dt>
        </dl>
```

dt と dd タグは記述リスト、説明リストと呼ばれるHTMLタグ

```
for shopinfo in shopinfo_list:
    # URLから店舗情報を取得
    res = requests.get(shopinfo['specurl'])
    soup = BeautifulSoup(res.text, 'html.parser')

    # dl.p-shopDetails > dt/dd構造でdtが項目、ddが値になっている。これを辞書形式にする
    shopspecs = {}
    for dt, dd in zip(
        soup.find('dl', class_='p-shopDetails').find_all('dt'),
        soup.find('dl', class_='p-shopDetails').find_all('dd')
    ):
        # 値に 改行や空白文字があるので取り除く
        shopspecs[dt.text] = replace_str(dd.text)

    # 店舗情報をマップ情報に追加
    shopinfo.update(shopspecs)
    # INFO:ここにランダム待機時間があるとよさそう
```

最終的にできるデータ

```
>>> from pprint import pprint
>>> pprint(shopinfo_list)
[{'TEL': '0544-27-0004',
 'specurl': 'https://umya-yakisoba.com/shop/3776/',
 'お店名ふりがな': 'あき',
 'エリア': 'まちなか',
 '住所': '富士宮市野中東町112-1',
 '受入人数': '12',
 '営業時間': '10:00-21:00',
 '地図': 'B6',
 '定休日': '火曜日',
 '店名': 'お好焼 あき',
 '料金目安': '350~600円',
 '業種': '飲食店',
 '焼き方': 'お店',
 '調査員おすすめメニュー': 'キムチとチーズ入り',
 '調査員が見た特徴': 'キャベツとネギが多めに入っている',
 '駐車場': '4'},
 ...]
```

上のコードの注意点

※: この例ではサイトのページネーションに対応していません。

ページネーションについては資料のコードで対応しています。

※: この例を使って、何度もWEBスクレイピングはしないようにしましょう。

randomモジュールやtimeモジュールを組み合わせてランダム時間待機します

```
import random
from time import sleep

def random_sleep(a: int,b: int) -> None:
    """
    aからbまでのランダムな時間を持つ

    """
    time.sleep(random.randint(a,b))

# ランダム時間待てるようにする
random_sleep(2, 5)
```

データを読む/取り込む



データを加工/出力する



データを使う/活用する

データを加工/出力する

マップのもとになるデータを作成します

- **情報を整理して表形式ファイルで書き出す**
- [付録]地理情報を集める

情報を整理

どのフォーマットで書き出すか？

よくある地理データ構造、フォーマット形式

- CSV（区切り表形式、汎用性高）
- GeoJSON（WEB APIで広く流通しているJSON形式の地理情報向け）
- KML（XML形式）

CSVライブラリを使って書き出せます

`csv.DictWriter` を使うとdictのkeyを使って列見出しを用意できる

```
import csv

# ※これは実は動きません
with open('mapdata.csv', 'w', newline='') as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
    fieldnames = list(shopinfo_list[0].keys())
    writer.writeheader()
    for shopinfo in shopinfo_list:
        writer.writerow(shopinfo)
```

各それぞれのお店情報の項目名が違う 😞

```
>>> shopinfo_list[0].keys()
dict_keys(['specurl', '店名', 'エリア', 'お店名ふりがな',
'地図', '業種', '住所', 'TEL', '営業時間', '定休日', '受入人数',
'駐車場', '焼き方', '料金目安', '調査員おすすめメニュー', '調査員が見た特徴'])
>>> shopinfo_list[2].keys()
dict_keys(['specurl', '店名', 'エリア', 'お店名ふりがな',
'業種', '住所', 'TEL', '営業時間', '定休日', '受入人数', '駐車場', '料金目安'])
>>>
```

→ 全部対応した項目を用意して列の見出し名（ヘッダー名）を作る

```
with open('mapdata.csv', 'w', newline='') as csvfile:  
    # フィールド名がまばらだったので、生成する  
    # すべてshopinfoのキーからフィールド名を取得してsetで重複を取り除いてリスト化  
    fieldnames = list(set().union(*shopinfo_list))  
  
    # 上のコードを丁寧に書く  
    # all_fieldnames_by_shopinfo = (list(shopinfo.keys()) for shopinfo in shopinfo_list)  
    # fieldnames = list(set().union(*all_fieldnames_by_shopinfo))  
  
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)  
    writer.writeheader()  
    for shopinfo in shopinfo_list:  
        writer.writerow(shopinfo)
```

出力できたCSVファイル

mapdata.csv

1 住所,営業時間,使用麺,ホームページ,駐車場,焼き方,調査員おすすめメニュー,TEL,エリア,
ア,お店名ふりがな,受入人数,specurl,地図,店名,業種,定休日,調査員が見た特徴,料金目
安
2 富士宮市野中東町112-1,10:00～21:00,,,4,お店,キムチとチーズ入り,0544-27-0004,まち
なか,あき,12,<https://umya-yakisoba.com/shop/3776/>,B6,お好焼 あき,飲食店,火曜日,
キャベツとネギが多めに入っている,350～600円
3 富士宮市元城町13-20,7:00～19:00,,,なし,お店,塩焼そば,0544-22-3029,まちなか,あさ
や,10,<https://umya-yakisoba.com/shop/3777/>,C4,あさ家,飲食店,なし,具たくさん大
盛、朝定食あり,400円～
4 富士宮市淀師523-4,11：00～21：00（休憩14：00～17：00）,,,10,,,0544-27-6005,郊外
・北部,あるばとろす,30,<https://umya-yakisoba.com/shop/3774/>,あるばとろす,飲食
店,月曜日（第4週日・月連休）,,500円～
5 富士宮市宮町2-9,平日11:00-16:00土日祝11:00-17:00,曾我麵,,なし,お店,焼きそば、ピリ
辛焼きそば,0544-26-3849,まちなか,いっぷくてい,20,<https://umya-yakisoba.com/shop/3616/>,B4,いっぷく亭,鉄板焼・軽食,月曜・第3日曜,麵はもちもちでコシのある歯ごたえ
が好評で、野菜は細く切り麺とソースと絡んで食べやすく,550～750円
6 富士宮市淀師468-2,10:30～17:00※材料がなくなり次第終了,,[https://tabelog.com/
shizuoka/A2204/A220401/22002448/](https://tabelog.com/shizuoka/A2204/A220401/22002448/),25,お店,五目焼きそば,0544-27-6494,まちなか,いと
う,50（3部屋あり）,<https://umya-yakisoba.com/shop/3772/>,A1,お好み食堂伊東,飲食
店,日曜・火曜 大きな鉄板でアツアツをどうぞ,550～850円

データを使う/活用する

旅行中に使うためのツールとして

- **巨人の肩に乗る: Googleマイマップで使おう**
- [付録]ポータブルに扱う: 印刷をする
- [付録]専用のWEBアプリを作ろう

Googleマイマップで使おう

- Googleが提供するマップのピンや経路をオリジナルで作成できるサービス
- アカウントに紐づいて、スマホ版Googleマップでも表示可能

扱い方

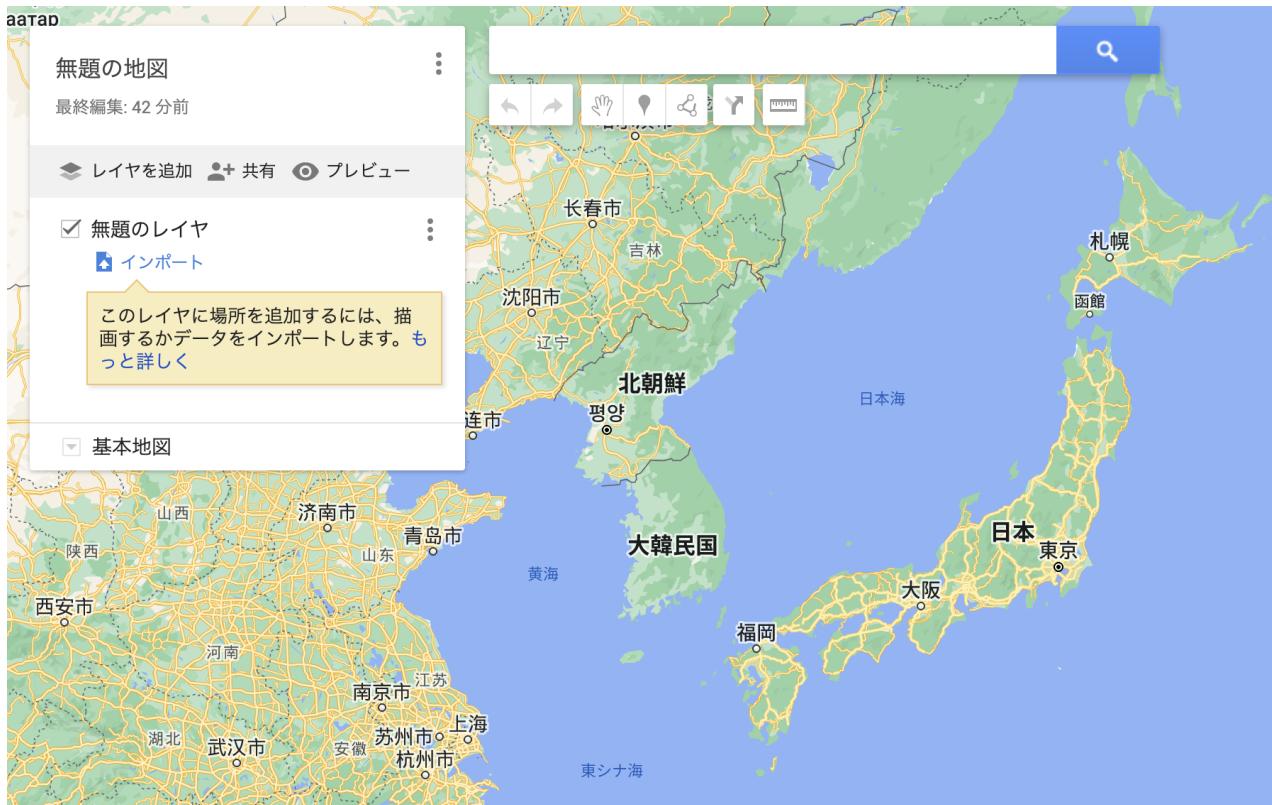
<https://www.google.com/maps/d/> ヘアクセスして利用します。

「新しい地図を作成」ボタンを押す

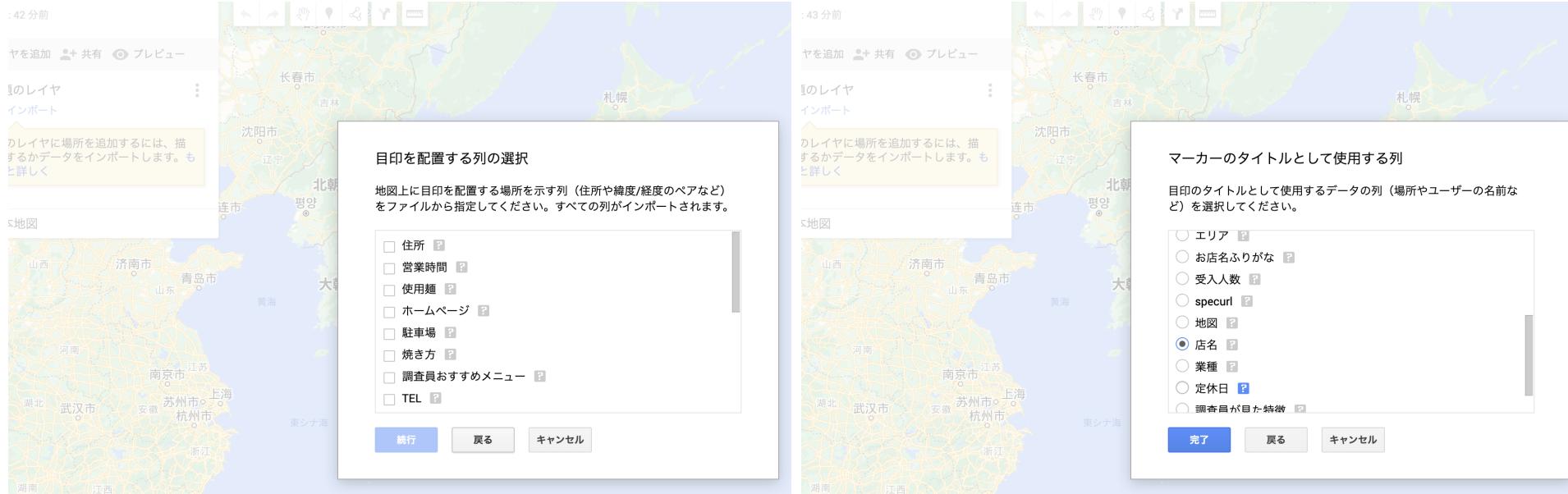
≡ Google My Maps



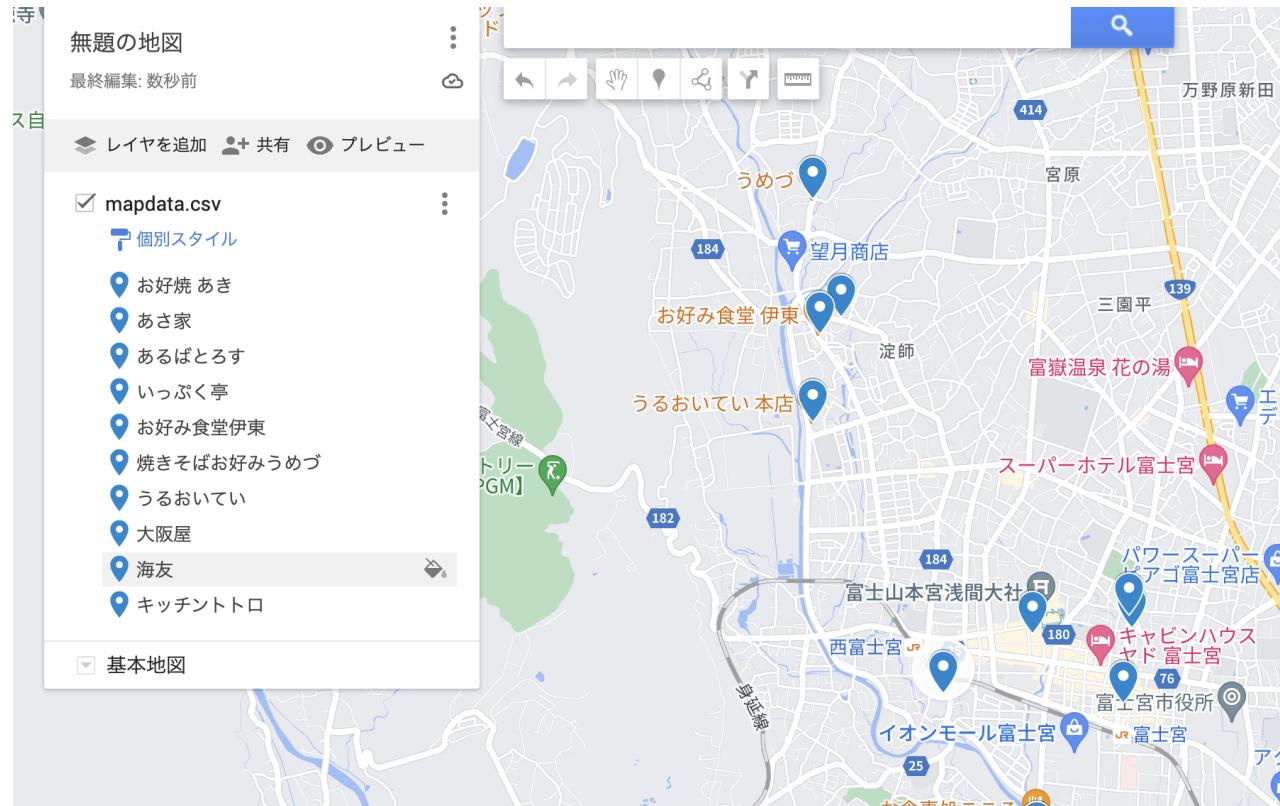
新規レイヤーへインポートする



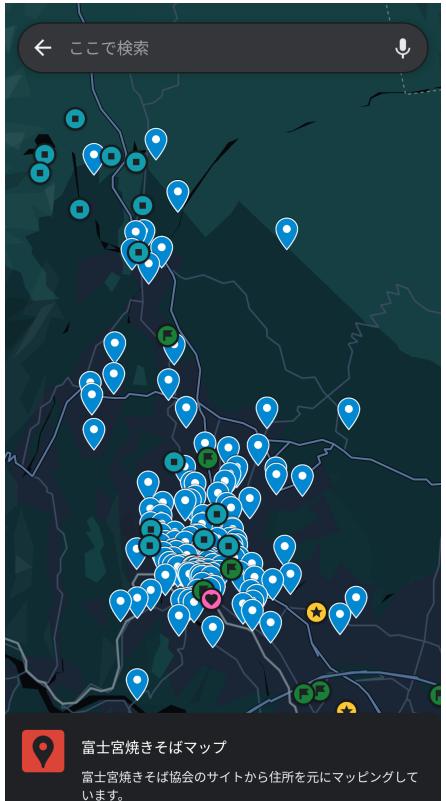
(操作していく)



マッピングされました!



やっぱりスマホで使えると便利！ナビも使える🚗



そのほかの選択肢

今回はGoogleマップを利用しましたが、推奨しているわけではなくてデータを作るいろんなサービスと連携できることをお伝えしました。

※: オリジナルのマップを作るサービスは他にも多数あります。一例を載せておきます。

※: それぞれ特徴や無料有料があるので、使いやすいものを探すと良いと思います

- OpenStreetMap uMap: <https://umap.openstreetmap.fr/ja/>
- proxi: <https://www.proxi.co/>
- [日本向け]国土地理院: <https://maps.gsi.go.jp/>
- etc...

トークのまとめ

データを読む/取り込む



データを加工/出力する



データを使う/活用する

今回のトークで目指すこと

- お店情報をWEBスクレイピング
- 表形式に整形 -> CSVファイル
- Googleマイマップで呼び出す



- データを読む/取り込む
- データを加工/出力する
- データを使う/活用する

今日学んだことを応用すると

- WEBスクレイピングでデータ収集 -> 世の中のWEBサイトの収集
***扱いには気をつけましょう**
 - WEB APIを扱えると収集や操作はもっと楽
- データの書き出し: CSV以外にもjson, xlsx, etc....
- 他のサービスへの連携: データベースに入れたりもできますよね

プログラミングでもっとも行うこと

- データを取り出す
- データを書き出す

- どこから「データを取り出す」か
 - データベースから使う
 - データ分析で現実の統計データを使う
 - センサーデータで現実環境を使う
- どこへ「データを書き出す」か
 - ユーザー情報をもとにレコメンデーション
 - データ分析でサービス改善
 - 得られたデータを使って製造工程改善 -> 業務改善にも

これができると、仕事も人生まで改善できますよ！

伝えたいこと

PyCampやPythonの基礎を学んだ方の一歩先として！

自分が使いたい、利用したいものやことでトライする。

できたらとても楽しいし感動します。オススメです！

Happy Hacking!!

and, Have a nice trip!!

付録

今回扱わなかった他の方法については、またどこかで解説できたらと思います

- WEBスクレイピングしづらいサイトもカバーする
 - selenium + headless chrome
- 画像識別でお店情報を収集する
 - OCR, Googleなど
- 緯度経度を収集
 - 世界: Google?
 - 日本: 東大CSIS
- 印刷物を作る -> テンプレートエンジンで印刷しやすいHTMLを生成
- fletで自分専用のマップアプリを作る

WEBスクレイピング: Seleniumの例

- 動的な（javascriptなど利用した）サイトはrequestsで対応が難しいことがあります
- 本物のブラウザ + ブラウザ自動操作ツール(selenium)を使った例も載せておきます

-> url

画像識別

結構難しい分野（ちょっと自信ない）

- Google Cloud Vision
 - -> url

位置情報

- 世界: Google Maps Platform ->
<https://developers.google.com/maps/documentation/geocoding/overview?hl=ja>
 - -> url
- 日本: 東大CSIS -> jageocoder
<https://github.com/t-sagara/jageocoder>
 - 無料で利用できる（リクエスト数は少なめがよし
 - -> url

印刷用マップを作ってみる

印刷用のHTMLファイルを作って印刷してみる

- Mapboxで概要と詳細の地図を用意
- お店の情報をテーブルで埋め込む
- 印刷用にCSSで調整する

-> url

モバイル向けのWEBアプリ

実験的な扱いです。うまく動かなければ...

- fletを使って作ってみました
- staticなページでWASMとして埋め込んで作ってみる
 - <https://flet.dev/docs/guides/python/publishing-static-website/>
- ※サイト自体の公開はしません

-> url