2022
春号

百萬石
Hyakumangoku Vol.58

Published by

部長挨拶

terry

terry@mma.club.uec.ac.jp

「百萬石」をお手に取っていただき，ありがとうございます．弊誌 (以下，部誌) は，電気通信大学の公認サークル MMA が新歓期 (春号) と弊学が開催する調布祭時期 (秋号) の年 2 回発行・頒布しているものです．MMA とは，”Microcomputer Making Association”の頭文字に由来しており，1975 年に創立されたサークルです．現在では計算機に関連することを中心に，ネットワークやセキュリティなどの分野でも活動しています．部誌には，MMA の部員が各々に行なっている活動を記事にしたものが掲載されています．内容は多岐に渡り，高度なプログラミングの話から旅行記など，MMA の活動の自由さが感じられるものとなっております．

さて, この「春号」は新歓期間中に配布されるので, きっと多くの新入生の方に御覧頂いていることかと思います．そのため，ここからは少しだけサークルの特徴をご紹介したいと思います．

MMA は技術に強い人材がたくさんいます．しかし，**初心者お断りサークルでは決してありません**．そういった先輩や仲間に支えてもらいながら，自分の好きなこと，やってみたいことを実現していけるかもしれない魔法のようなサークルです．計算機 (コンピュータ) を使っていさえすれば，活動内容は自由です．ただ，このメリットは「やることを探さなくてはならない」というデメリットと表裏一体です．入ったはいいものの，なにをすればいいのかわからないがために，幽霊部員となっていく人は多いです．特にコロナ禍という状況下ではより一層その傾向が強まったと思います．

そこで我々部員は「新入生講習会」を用意しました．ここで，先輩との接点を持ち，雑談のような会話をしていく中で，自分の興味はどこにあるのかを探してもらい，やりたいことを見つけてもらおうと思っています．

これまでは部活の抽象的な特徴についてお話しましたが，具体的な活動内容を紹介します．部員全員が部室に出入り自由になるような電子錠システムの開発 (ソフトウェア開発)，ある制約の中での課題について，答えを導くプログラムを正確に作る速さを競う競技プログラミングの大会に参加することや，部内勉強会の開催などを行っています．

最後に，MMA はいろんな人とのつながり，自分のやってみたいことを相談できる環境，充実した機材が揃っていると自分は感じます．そんな自由度の高いところで活動している部員が書く部誌は面白いものばかりだと思います．ぜひご一読ください．

2022 年 4 月吉日

百萬石 Vol.58

Hyakumangoku Vol.58

2022 Spring

MMA

目次

第 1 章　Markdown からイケイケスライドを作るのを自動化して最強になった　terry　1
1.1　導入：スライド，きれいに見せたくない？ 1
1.2　前段：さらば Beamer，ようこそ Marp 1
1.3　本題：Marp 使いやすすぎ！ 3
1.4　発展：GitLab CI/CD を用いた自動化の話 3
1.5　総括：まとめと感想 4

第 2 章　6 秒に 1 回 Twitter を監視するプログラムを書いた　fjdj　5
2.1　はじめに 5
2.2　作ったもの 5
2.3　どんな感じで動かしているか 8
2.4　おわりに 9
参考文献 9

第 3 章　古いノート PC に AlterLinux を入れて、ローカル GitLab サーバーを動かしてみた　azarasing　10
3.1　はじめに 10
3.2　AlterLinux 導入 11
3.3　gitlab サーバーの構築 13
3.4　まとめ 14
参考文献 15

第 4 章　関東ビッカメ娘往訪記　maccha　17
4.1　はじめに 17
4.2　往訪記 17
4.3　さいご 21
参考文献 21

第 1 章

Markdown からイケイケスライドを作るのを自動化して最強になった

terry

概要

ダサい見た目の PowerPoint から卒業すべく，Markdown からスライドを生成した．この時点で最高なのに，Markdown を push するだけで自動でスライド PDF が帰ってくるようにした．最強！という記事．この記事は `https://www.mma.club.uec.ac.jp/~terry/posts/auto-marp-pdf/`(個人ブログ) を部誌用に少しアレンジして書き直したものです．そのため本質的に同一内容です．[*1]

1.1 導入：スライド，きれいに見せたくない？

PowerPoint のスライド，**ダサい**と思うことはありませんか？

僕はあります．WYSIWYG のやつはマウスと戦わなきゃ行けなくてしんどいんですよね．ところで，WYSIWYG はコンリテ[*2]でやってるからもちろん皆さんご存知ですよね？見た目をいじらなくていい場合，ある程度完成された見た目になる傾向があると僕は思っています．LaTeX とかがいい例で，何もしなくてもある程度美しいじゃないですか．今流行でちょっとした物書きといえばの **Markdown** でスライド作れないかな？と思いました．できるようにするために戦ってたのでその過程を書きます．最終的にはこんな感じ (図 1.1，次ページ) のスライドができます．

1.2 前段：さらば Beamer，ようこそ Marp

Beamer，数式メインくらいならいいんですが，そういう資料よりもカジュアルだったり数式少なめなものには向かないかなぁと思ってしまいました．何より記述量が多いので避けたくなりました．

[*1] この個人ブログも見てね！

[*2] 新 1 年生は授業で前期のうちにやるよ！

図 1.1　できたスライド

1.2.1　Beamer と Pandoc との別れ

pandoc から markdown を Beamer という LaTeX 記法のスライド出せるやつに変換できれば嬉しいなと思ったんですよ．Markdown の方が LaTeX より記法楽だしね．

でも上手くテーマやフォントがはまらなかったりして，めんどくさくなってしまいました．変換後のソースをいじるの，きれいなソースがでてこないから苦痛でした．思い通りにならなくてしんどかったのでやめました．

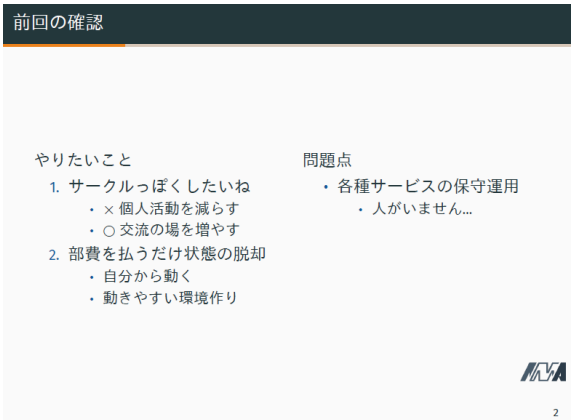

図 1.2　markdown から変換した Beamer スライド (Metropolis)

もうこれ Markdown を直接スライドにしたくね？となったのです．あと普通に Beamer だと CSS がいじれるわけじゃないから見た目のカスタマイズがやりにくすぎる． メリットでもあるけどデメリットでもあるのよね．

というわけで Markdown でいい感じのスライドが作成できるツールを探す旅に出た．

1.2.2 ある記事と Marp との出会い

みんなのお友達 Google さんで「Markdown スライド」って戯れてたら神記事が降ってきました．経緯は省略 (続きは Web(個人ブログの該当ページ) で)

1.3 本題：Marp 使いやすすぎ！

1.3.1 導入：Marp for VSCode 最強！

ココ[*3]を見ながら VSCode に拡張機能入れただけで Markdown 書けばすぐスライド！神すぎ！

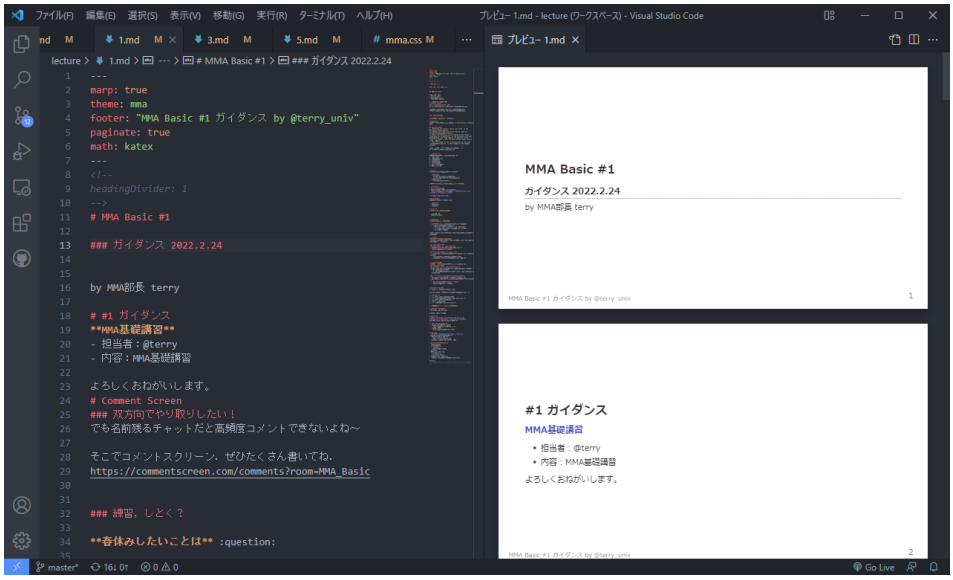

図 1.3 Markdown 書くだけでスライドできるってマジ？

1.3.2 多少のトラブルに遭遇

自作テーマ (CSS) を使わなければ問題なし，うまく行かないトラブルに出会ったらもしかしたら解決策が僕の個人サイトにあるかも？ (`www.mma.club.uec.ac.jp/~terry`)

1.4 発展：GitLab CI/CD を用いた自動化の話

GitLab CI/CD は GitHub Actions と同じようなものです．なぜ GitLab かというと MMA は部室で自分たちでホスティングした GitLab を持っているのでということで GitLab を採用．

[*3] 【VS Code + Marp】Markdown から爆速・自由自在なデザインで、プレゼンスライドを作る/Qiita 記事参照

「感想：これ GitLab のほうが書きやすい」

1.4.1 手順というか構造というか

1. GitLab にデータを push
2. GitLab CI/CD(GitHub Actions) を使って Marp-CLI を動かす
3. Artifacts に成果物残して嬉しいね！ (完成)

Marp-CLI は node.js[*4]があれば動きます．そのため node がインストールされてる Docker 引っ張ってきてスクリプト書くだけです．以下参考までに．(これは Web に全部は乗ってない) フォントあたりの設定に苦労しましたがココでは省略 (続きは Web で)

ソースコード 1.1 .gitlab-ci.yml

```
generate-pdf:
    stage: build
    image: node:latest
    variables:
      http_proxy: http://proxy.uec.ac.jp:8080
      https_proxy: http://proxy.uec.ac.jp:8080
artifacts:
    paths:
      - ./*.pdf
script:
    - apt update
    - apt install -y chromium task-japanese locales-all # 日本語対応
    - npx @marp-team/marp-cli@latest  ?.md --pdf --allow-local-files --theme
        ./themes/mma.css # md to pdf
```

1.5 総括：まとめと感想

まとめると，Marp を使うとワンライナーでスライド PDF を作れるぞ！それなら GitHub 系サービスにやらせてみると便利じゃね？って記事でした．

部誌に `terry` は初めて技術系の記事を書きました．部長としてどうなん？感ありますがやっとですよ．やっと．今までどんな記事を書いてきたかと言うと，コロナ禍のサークル，授業で使える Tool の話と技術遠め大学多めでした．この記事の内容を使うとかなり便利になりまして，私が実施する講習会はこれでスライド作っていくことにします．最高だ〜〜！

過去の部誌は調布祭ページ[*5]で全部読むことができます．結構面白いと思うのでぜひ読んでね．

MMA の雰囲気を掴み取ることはできましたか？私は技術ですがネタに走った人もいます．ぜひぜひこの自由なサークルで一緒に大学生活楽しみましょう！私は窓口開けとくので MMA の質問ならなんでも気軽にどうぞ！

入部，待ってるよ．

[*4] 最近知ったんだけどこれ javascript 実行環境らしい．バックエンドの何かではないらしい (よくわかってない)

[*5] `https://www.mma.club.uec.ac.jp/chofu-festa/`

第 2 章

6 秒に 1 回 Twitter を監視するプログラムを書いた

fjdj

2.1 はじめに

2.1.1 自己紹介

はじめまして、21B の fjdj と申します。部誌を書くのは初めてとなります。ネタはいくらでも転がっているのですがどうしてもやる気が出なくて重い腰を上げてこの文を書き始めたところです。さて、UEC で最初に入ったサークルが MMA なのですが、気づいたら無線部 (JA1ZGP) と調布祭実行委員会、さらにはネットワーク管理委員会に入っていました。楽しいですね。周りからは過労死を心配されてるらしいですがまだ倒れてないので大丈夫だと思います。

2.1.2 きっかけ

さて (2 回目)、所属しているとある団体では Twitterer が多かったり少なかったりするのですが、その中にはツイート数が 3 万を超えている人もいます。別に何をどれだけツイートしようがかまわないのですが、たまに仕事をしていないのにツイートを連投しているときがありまして、これはけしからんなと思ったわけであります (登大遊ボイス)。そこでその人の Twitter を監視して新しいツイートがあったら即座に Discord に飛ばして通知を鳴らして驚かせてやれば仕事をしてくれるのではないかと思ったわけであります。幸い、高校のときに (も) ヘンなプログラムを書いたことがありまして、TwitterAPI を使えるアカウントを持っていましたのでこれを活用することにしました。

2.2 作ったもの

できあがったものがこちらです。

2.2.1 プログラム

ソースコード 2.1 作ったプログラム

```
from requests_oauthlib import OAuth1Session
import requests
import datetime
import json
import pandas as pd
import sys
import os
import time
nbm = dict.fromkeys(range(0x10000, sys.maxunicode + 1), '') #
    non_bmp_map
import re

API_KEY = os.getenv('API_KEY')
API_SECRET = os.getenv('API_SECRET')
ACCESS_TOKEN = os.getenv('ACCESS_TOKEN')
ACCESS_SECRET = os.getenv('ACCESS_SECRET')

url = "https://api.twitter.com/1.1/statuses/user_timeline.json"
params ={'user_id' : sys.argv[1]}

dir_path = os.path.dirname(os.path.abspath(__file__))
csv_path = "%s/csv/%s.csv" % (dir_path, sys.argv[1])
csv_charset = "utf-8"

webhook_url = os.getenv('DISCORD_WEBHOOK_URL')

def cleanhtml(raw_html):
    cleanr = re.compile('<.*?>')
    cleantext = re.sub(cleanr, '', raw_html)
    return cleantext

def pushToDiscord(sname,tweetid,icon):
    main_content = {
      "content": "https://twitter.com/%s/status/%s" % (sname, tweetid),
      "avatar_url": icon
    }
    requests.post(webhook_url, main_content)

def get_tweets():
    # OAuth
    twitter = OAuth1Session(API_KEY, API_SECRET, ACCESS_TOKEN,
        ACCESS_SECRET)
    time.sleep(1)
    req = twitter.get(url, params = params)
    arr = []

    if req.status_code == 200:
        timeline = json.loads(req.text)
        for status in timeline:
            time1 = str(status["created_at"])
            time2 = datetime.datetime.strptime(status["created_at"], "%a
                %b %d %H:%M:%S %z %Y")
            time2 = (time2 + datetime.timedelta(hours=9)).strftime("%Y/%
                m/%d %H:%M:%S")
            tweetid = str(status["id"])
            userid = str(status["user"]["id"])
            name = status["user"]["name"].translate(nbm)
            sname = status["user"]["screen_name"].translate(nbm)
```

```
                source = cleanhtml(status["source"])
                text = status["text"].translate(nbm)
                geo = status["geo"]
                coordinates = status["coordinates"]
                place = status["place"]

                userloc = status["user"]["location"]
                userdes = status["user"]["description"]
                userurl = status["user"]["url"]
                userprotect = status["user"]["protected"]
                followers = status["user"]["followers_count"]
                follows = status["user"]["friends_count"]
                lists = status["user"]["listed_count"]
                created_at = status["user"]["created_at"]
                favourites = status["user"]["favourites_count"]
                tweets_counts = status["user"]["statuses_count"]
                profile_image = status["user"]["profile_image_url_https"]
                profile_background = status["user"]["
                    profile_use_background_image"]

                arrrr = [time1, time2, tweetid, userid, name, sname, source,
                     text, geo, coordinates, place, userloc, userdes,
                    userurl, userprotect, followers, follows, lists,
                    created_at, favourites, tweets_counts, profile_image,
                    profile_background]
                arr.append(arrrr)

                time2c = datetime.datetime.strptime(time2, "%Y/%m/%d %H:%M:%
                    S")
                dt_now = datetime.datetime.now()
                if dt_now - time2c < datetime.timedelta(minutes = 0,seconds
                    = 6):
                    pushToDiscord(sname,tweetid,profile_image)

        else:
            print("ERROR: %d" % req.status_code)

        arr.reverse()
        return(arr)


    def main():
        os.makedirs("%s/csv" % dir_path, exist_ok=True)
        arr = get_tweets()
        df = pd.DataFrame(arr, columns=['time','localTime','tweetId','userId
            ','screenName','userName','source','text', 'geo', 'coordinates',
             'place', 'userloc', 'userdes', 'userurl', 'userprotect', '
            followers', 'follows', 'lists', 'created_at', 'favourites', '
            tweets_counts','profile_image', 'profile_background'])

        if not os.path.exists(csv_path):
            df.to_csv(csv_path, encoding=csv_charset, index=None)
        else:
            df_0 = pd.read_csv(csv_path, encoding=csv_charset)
            df_out = pd.concat([df_0, df])
            df_out = df_out.drop_duplicates(['tweetId'])
            df_out.to_csv(csv_path, encoding=csv_charset, index=None)

    if __name__ == "__main__":
```

```
        main()
        endtime = datetime.datetime.now()
        print('%s done' % endtime)
```

まぁ、GitHub にある[*1]ので載せる必要はないのですが載せてみました。プログラミングはあまり得意でないので汚い書き方ばかりですが、動けばよいのですよ、動けば (ホンマか？)。

プログラムは高校の頃から使っている Python で書きました。Python は脳死で書けるのでよいですね。でも、簡単な言語から入ったせいで基礎プロでは C にちょっと苦戦してしまいました。

プログラムの説明としては、TwitterAPI の user_timeline を叩いて最新のツイートを取得、取得したツイートの作成時間が 6 秒以内であれば Discord の webhook に投げるというシンプルな仕組みです。あと、ついでに面白そうな情報を配列に入れて csv に保存する機能も付けてみました。これに関しては pandas がすごくて、`df_out = df_out.drop_duplicates(['tweetId'])` で tweetId が重複している行を消してくれます。昔 GAS でスプレッドシートを触っていた[*2]ときは、if 文で無理やりやっていたこともありました。とはいえこのコードももっといい方法がありそうですが…

2.2.2 実行スクリプト

これまでの自分の知識では crontab だけを駆使してやっていたと思うのですが、MMA には slack-kagisys という優秀なシステム (プログラム) がありまして、その実行スクリプトを参考にさせていただきました。

ソースコード 2.2 実行スクリプト

```
#!/bin/sh

while true
do
/home/fjdj/.local/bin/pipenv run python notice-user-tweet.py ${userid} 2>> /
    home/fjdj/twitter-analysis2/err.log > /dev/null;sleep 5;
done
```

while を使った単純なシェルスクリプトですが、こういうやり方があるのだと初めて知りました。

2.3 どんな感じで動かしているか

家に自作の Linux サーバーがあるのでそこで実行しています。

ところで、TwitterAPI には当然回数制限がありますがこの v1 の user_timeline に関しては 900 Requests/15-min window (user auth) だそうです。6 秒間隔だと 15 分で大体 150 回くらいですから、6 秒間隔なら最大 6 アカウントのツイートを監視することができます。ヤッタネ！ (そんなに使うことあるか？)

[*1] 今回環境変数を使うなどして初めて GitHub で Public にできるようにしました
[*2] 学校の HP の更新通知とか、気象情報の収集とかやってました

2.4 おわりに

いかがでしたか？ (え)

ここまでダッシュで書きましたが全然書いてないことたくさんあるな〜と思いました。pipenv とか昔作った GAS の話とか、自宅サーバーの話[*3]とか…また機会があったら書こうかと思います。

ここまで読んでくださりありがとうございました。

参考文献

[1] https://github.com/fjdj-1515/twitter-analysis2

[*3] 現在は Ubuntu な自作デスクトップ 1 台と RaspberryPi が 3 台あります

第 3 章

古いノート PC に AlterLinux を入れて、ローカル GitLab サーバーを動かしてみた

azarasing

3.1　はじめに

3.1.1　自己紹介

どうも azarasing(あざらし) です。久しぶりに部誌を書こうと思います。調布祭実行委員会であれやこれやしていましたが、最近ようやく (4 年間近にして) 暇になったので、隙間時間で部誌を書いています。(間に合うかな・・・) ツイッターは@saitoshiryu という昔のリテラシーのへったくれもないアカウントを使ってます。良ければフォローおば。

3.1.2　AlterLinux とは？

AlterLinux とは、パソコンに組み込む OS の一つという解釈でいいです。Windows とか Mac とかいうあれです。その一つとして無料で公開されている Linux と一般に呼ばれる OS 群の中の一つが AlterLinux です。なぜ、わざわざ OS を変えるのかということですが、Windows をはじめとした大手 OS はその機能の多さ故、無駄な機能が多くスペックの小さい PC では動作が重くなることもしばしばです。しかし OS を変えることでその動作を制限した上で軽くすることができるという認識で問題ありません。

このような OS の中でも AlterLinux は Arch 系と呼ばれる系統の一つであり、電通大一年生が必ず使わされる Ubuntu という OS とは異なる系統の OS となります。この手の OS は Ubuntu と違った操作性があり、経験しておくことは重要です。その Arch 系の一つである ArchLinux は導入のハードルが高いイメージがありますが、AlterLinux はこんな Linux についてふんわり解説している筆者でも簡単に入れることができるものになっています。また日本の大学生集団がつくっている Linux なので、日本語サポートが充実しています。なので、初心者には Ubuntu の次にオススメな Linux ディストリビューショ

ンといえるでしょう。

3.1.3 GitLab とは

Git という分散型ソースコード管理システムの一種です。これを使うと仲間内のレポートを共同で書いたり、レポートについて実際の作業を参照しながら議論したりできます。GitLab のサービスは GitLab の本部 (恐らく海外であると思われる) にあるサーバーを使い、無料でサービスを受けることができますが、レポートなどの情報を海外のサーバーに置くという行為があまり推奨されないことは理解した方がいいです。こういった懸念をなくすために、自宅にサーバーとなる端末を設置してそこで管理するようにします。

3.1.4 注意!!

ここに書いてある方法はあくまでも個人の環境に沿った方法であり、ここに書いてある方法を参考に自身の PC を変更した際に生じた損失は負えません。試す際はあくまでも自己責任でお願いします。

3.2 AlterLinux 導入

3.2.1 導入 PC スペック

製品名: Lenovo G510
CPU: intel corei5-4200N (2.50GHz)
メモリ: 8GB (DDR3)
容量:SSD(500MB:boot) HDD(1TB)
※ 古いので、ノート付属のキーボードが使えませんでした ww

3.2.2 ブート USB を作る

OS をインストールするためにはそれを起動するためのブート USB を作り、インストールする必要があります。

AlterLinux では、このページ[*1]から iso ファイルをダウンロードしましょう。

AlterLinux にはいくつかのエディションがありますが、ここでは軽量で汎用的な XFCE を選択しましょう。これをダウンロードしたら、もう使わない USB メモリを用意します。ブート USB を作るときの注意点ですが、ブート USB となった USB は元の状態に戻すことはできないため、USB メモリとしてはもう使えなくなってしまいます。その辺を考えて、使い捨ての USB を用意することが必要です。

USB にイメージファイル (iso ファイルのこと) を書き込むためには、ただ iso ファイルを USB メモリに移すだけの作業では意味がなく、それが OS のブートが行える USB であることを書き込む必要があり

[*1] https://alter.fascode.net/downloads/

ます。そのため、balenaEtcher という別のソフトウェアから書き込みます。ダウンロード先はこちら[*2]です。

balenaEtcher を起動したら、SelectFile でダウンロードした iso ファイル、target で自分の PC に挿している USB を選択して DD モードで書き込みを行います。これを完了すると、USB をフォーマットする必要がありますとでてきますが、無視して USB を抜き取り、自分が AlterLinux にしたい PC に挿して PC を起動します。

3.2.3　AlterLinux のインストール

Windows10 の場合は、shiftを押しながら再起動ボタンを押すと、起動オプション画面に行くことができます。そこから、デバイスの使用を選択し、USB メモリを選択します。これを起動すれば、正常に USB から AlterLinux が起動するはずです。

AlterLinux が起動したら、USB から本体へ AlterLinux をインストールしていきましょう。デスクトップにある Alter のマークのアプリを開いて、本体への書き込み設定を行いましょう。ここら辺の設定は適当で問題ないですが、できるだけ推奨とされているものを選びましょう。

内部への PC への書き込みが正常に完了したら、USB を抜いて内部の HDD もしくは SSD にて AlterLinux が起動していることを確認しましょう。

[*2] https://www.balena.io/etcher/

3.3 gitlab サーバーの構築

さて、起動した AlterLinux のアプリケーションから、ターミナルを起動しましょう。まずは、パッケージを最新のものに更新する必要があるので、

```
sudo pacman -Syu
```

と入力して、パッケージをインストールします。

gitlab は、OS の上に Docker という環境を載せて、そのうえで起動させます。つまり docker をインストールする必要があるので、まずはそれを行いましょう。

3.3.1 Docker のインストール

Docker のインストールも簡単に行えます。ターミナルから

```
sudo pacman -S docker docker-compose docker-machine
```

このように打ち、実行して Docker をインストールします。これが終わったら、以下のコマンドを打ち、Docker を再起動します（誤解を招く言い方）。

```
sudo systemctl enable docker
sudo systemctl restart docker
```

3.3.2 Gitlab のインストール

ここまでくれば、後は Docker にある Gitlab のサーバーを持ってくればいいだけになりました。docker をまとめて実行するために、YAML ファイル (docker-compose.yaml) を作成しておきましょう。

```
gitlab:
  container_name: gitlab01
  image: 'gitlab/gitlab-ce:latest'
  restart: always
  hostname: 'localhost'
  environment:
    GITLAB_OMNIBUS_CONFIG: |
      external_url 'http://192.168.0.192:12000'
      gitlab_rails['time_zone'] = 'Asia/Tokyo'
  ports:
  - '12000:12000'
  - '2022:22'
```

```
volumes:
- '/docker/gitlab/config:/etc/gitlab'
- '/docker/gitlab/logs:/var/log/gitlab'
- '/docker/gitlab/data:/var/opt/gitlab'
```

ここで、この external_url には自身の環境の IP アドレスである必要があります。ここでは、ip address と打ち込み事前に端末の ip アドレスを確認しておくのがよいでしょう。

この docker-compose.yaml があるディレクトリで、

```
docker-compose up -d
```

と打ち込むと、インストールが開始されます。

これが終われば、ローカルの gitlab にアクセスできます。別の端末でもいいので、http://192.168.0.192:12000 にアクセスして、gitlab のローカル環境にアクセスできることを確認しましょう。この時のアドレスは、環境によって異なります。

3.4　まとめ

AlterLinux をインストールして docker 環境を整えて gitlab を起動させてみた。

実際の運用で使うためには、さらにユーザ登録やポート開放などを行う必要があります。

えっ？メンドクサイ？

そんなあなたに朗報です。今なら MMA 部員になるだけで、gitlab の環境が使いたい放題になります！こんな面倒なことをしなくても諸先輩方が構築したつよつよサーバーの上で動く Gitlab が使えます。サーバーも電通大の部室にあるので安心して使うことができます！　皆さんも MMA 部員にならないか？

参考文献

[1] Gitlab を自分のパソコンや社内サーバで動かしてみる,https://blackbird-blog.com/gitlab-localinstall

[2] Alter Linux(Xfce 版) で Docker と GPU 対応の開発環境を作る, https://qiita.com/totto2727/items/910b1a1016704867311

[3] Pacman 比較表,https://wiki.archlinux.jp/index.php/Pacman/

[4] AlterLinux,https://alter.fascode.net/

図 3.1　iso ダウンロードページ

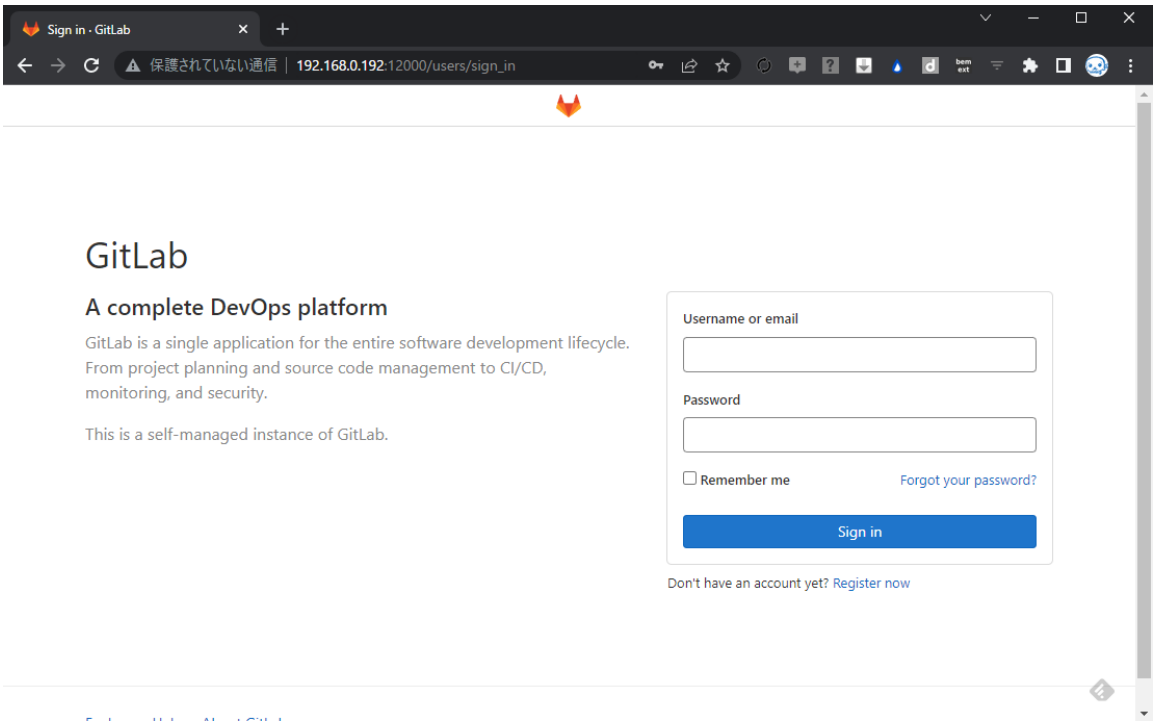

図 3.2　gitlab 起動ページ

第 4 章

関東ビッカメ娘往訪記

maccha

4.1　はじめに

私は新 B4 の maccha だ。入学して、あっという間に 3 年が経った。

2022 冬アニメは実に素晴らしかった。『明日ちゃんのセーラー服』は全 12 話を神作画で魅せてくれた。見るたびに、若さと瑞々しさを摂取できた。見るたび心が潤う。

『その着せ替え人形は恋をする』は、今期の覇権といっても過言ではないだろう。この作品も大満足の神作画で全話構成され、感涙にむせんだ。

両作品とも制作会社は CloverWorks である。CloverWorks さんは、過去に『青春ブタ野郎はゆめみる少女の夢を見ない』[*1]や『冴えない彼女の育てかた Fine』を制作した。来期は『SPYxFAMILY』や『くノ一ツバキの胸の内』を手がけるので、楽しみだ。

4.1.1　ビッカメ娘とは

2015 年、Twitter 上のナイセン公式アカウント (@itallinc) とビックカメラ水戸店 (@biccameramito) とのやり取りをきっかけに、ビックカメラ擬人化プロジェクトは始まった。水戸店の「みとたん」をはじめ、ビックカメラ各店舗の擬人化キャラクターは誕生し続け、現在もプロジェクトは進行中である。

4.2　往訪記

今回は、ビッカメ娘の等身大パネルが設置されている、関東のビックカメラを訪れた。条件を満たす店舗は 26 店舗。

等身大パネルは店内のどこかに設置されている。発見した等身大パネルは写真を撮った。また、各キャラクターの「名刺」が配布されている場合もあるので、見つけ次第回収した。

1 日目は azarasing くんが同行した。彼は過去にも、在来線で限界日帰り名古屋旅行[*2]を共にした戦友である。今回は少数精鋭で挑ませていただく。

[*1] アニメ 2 期待ってるよ

[*2] 東京・名古屋間を在来線だけで往復した。滞在は数時間で、長時間の移動の記憶がまるでない。狂ってる。

azarasing くんとは朝 6:30 に都内某駅で待ち合わせた。服からリュックまで黒く着こなした、一般黒づくめ男性の格好であった。

4.2.1 水戸たん

では、訪ねたビッカメ娘を往訪順に振り返る。カラーコード控えめな服装の azarasing と最初に向かったのは、水戸店だ。開店時間前に最寄りの水戸駅に到着した。

午前 10:00、駅直結のショッピングモール「エクセルみなみ」は開店した。4 階の入口脇に、「水戸たん」はひっそりと佇んでいた。水戸たんの名刺もあったので、喜んで頂戴した。

図 4.1 水戸たん

図 4.2 カメラマン (撮影 azarasing)

4.2.2 柏たん・ふなとーたん・ふなたん (船橋たん)

続いて、常磐線で柏駅へ。駅直結の「スカイプラザ柏」にビックカメラはある。「柏たん」は入口にいた。
ハッシュタグ「#柏たん」を添えて、この画像をツイートしたところ、ご本人からの反応を頂けた。
名刺は発見できなかった。どうやら店内にあったようだが、水戸店では名刺が等身大パネルと一緒にあったので、同じ場所にあるものだと思い込んでいた。

図 4.3 柏たんの反応

続いて船橋駅へ。

ふなとーたんは船橋東武店、ふなたんは船橋駅 FACE 店にいる。2 人の等身大パネルは、船橋東武店にあった。

船橋東武店は近々閉店するそうだ。ふなとーたんとのお別れはさみしい。お別れのメッセージカードがあったので、頂戴した。[*3]。

4.2.3 アキバたん・有楽町たん・みっけたん (赤坂たん)

秋葉原駅で降り、ビックカメラ AKIBA へ向かった。店内を探したが、アキバたんはなかなか見当たらなかった。どうにか発見できたが、アキバたんの名刺を手に入れ損ねた。不覚。

有楽町店でも、有楽町たんの捜索に苦労した。探偵 azarasing くんの協力のおかげで、見事見つけ出すことができた。なお、ここでも名刺は回収し損ねた模様。

赤坂見附駅店までの道のりは、azarasing くんに案内してもらった。赤坂見附駅店のみっけたんは、名前の割にはすぐには見つけられなかった気がする。髪留めが「坂」の形をしているのが、特徴。

4.2.4 しぶとーたん・しぶハチたん

渋谷の人混みをちぎっては投げ、ちぎっては投げ。渋谷東口店のしぶとーたんと、渋谷ハチ公口店のしぶハチたんに会いに行った。

4.2.5 川崎たん・新横たん・藤沢たん

渋谷から川崎まで、しばし移動。夕方に近づき、残す時間があまりない状態に。

川崎たんの名刺には制服のデザインが描かれていた。胸元にはツバキ、スカートは短冊のデザインであった。よき。

新横浜店は新横浜駅に隣接していた。新横たんは青髪のかわいい女の子。ボーイッシュみがいいよね。

藤沢駅まで、しばし移動。藤沢たんは片目を髪で隠したクール系。藤の花をモチーフにしたキャラデザみたい。よき。

4.2.6 さがみたん・町田たん

さがみたんはアジサイをモチーフにしている。他のビッカメ娘よりも年下という設定らしいので、ロリ専攻の方にはオススメ。

町田たんは眼鏡っ子で、ヘアピンと横に結んだ髪で「町」という字になる、芸の細かいことをしている。いいね。

[*3] 初めて出会ってから数秒でお別れ、お別れ RTA

4.2.7 八王子たん・立川たん

町田から横浜線で北上し、八王子駅へ。

八王子たんは、「王子」という名前から王冠を髪飾りにしている。制服のリボンは銀杏がモチーフ。よき。立川たんは萌え袖が特徴。横髪は立川名産のウド、胸元の花はコブシがモチーフ。大きい帽子とアホ毛も特徴だ。

図 4.4 立川たん

ここでビックカメラは閉店時間を迎えたので、本日はここまで。

4.2.8 せいせきたん・調布たん

2 日目はソロでアタックすることになった。

スタートは聖蹟桜ヶ丘駅店から。桜をモチーフにしているみたい。かわいい。

その次は京王調布店の調布たん。電通大の近くということもあり、知っている人は多そう。胸の花は調布市の花サルスベリがモチーフ。

4.2.9 ビックロたん・新東たん・新宿西口たん

続いて新宿駅へ。新宿駅周辺には目当てのビックカメラが 3 店舗ある。まず初めに向かってたのはビックロ ビックカメラ新宿東口店だ。

ビックロ ビックカメラ新宿東口店は、ビックカメラとユニクロが一体となった店舗である。ビックロたんの等身大パネルは 2 種類であった。また、この店舗はユニクロが併設されており、気のせいか服を見に来たカップルが多く、気圧された。しかし、私はここまで 18 名のビッカメ娘と交流しており、実質 18 人のビッカメ娘を侍らせている状況、つまり十八股と言っても過言ではない。私の方がより罪深いのだと思うことで、堂々と入店することができた。[*4]

その後は、新宿東口駅前店の新東たんと、新宿西口店の新宿西口たんを訪ねた。

[*4] MMA には抱き枕カバーに三十股している方もいるので、その点、私はまだノーマルである

4.2.10 池西たん・プロスタたん・パソ館たん (パソコン館たん)・本店たん

池袋駅周辺にはたくさんのビックカメラがあった。まずは池袋西口店。池西たんはおさげが特徴であった。
2 店目はビック・プロスタジオ 東京写真館。プロスタたんは写真館をモチーフにしたキャラクターであった。
3 店目はビックカメラ本店パソコン館。パソ館たんは首からスマホをさげていて、癖っ毛が特徴。かわいい。
4 店目は池袋本店。本館たんは袴のような制服を着て、頭にフクロウを乗せている。

図 4.5 池西たん

4.2.11 大宮たん

最後に訪れたのは、maccha の地元にある、大宮西口そごう店。店内には 2 種類の等身大パネルがあった。

4.3 さいご

ビッカメ娘を訪ねて、等身大パネルの写真をツイートする方は、よく見受けられる。タグ付きでつぶやくと、各店舗のアカウントが反応してくれることも多い。

ビッカメ娘がいない店舗はまだ数多く残っている。プロジェクトが進行し、今後関東で新たなビッカメ娘が生まれたら、ぜひ訪ねてみたい。

参考文献

[1] ”CloverWorks Official site”, CloverWorks Inc.,
`https://cloverworks.co.jp`
[2] ”ビッカメ娘 -ビックカメラ店舗擬人化プロジェクト-”, ビッカメ娘 プロジェクト事務局,
`https://biccame.jp`

百萬石 2022 -春- © 電気通信大学 MMA

2022 年 4 月 1 日　初版第一刷発行　【本書の無断転載を禁ず】

著　者 terry, fjdj, azarasing, maccha
表　紙 allocer
編集者 allocer
発行者　電気通信大学 MMA
発行所　電気通信大学 MMA 部室
〒182−8585 東京都調布市調布ヶ丘 1−5−1 サークル会館 208 号室
https://www.mma.club.uec.ac.jp/
印刷所　電気通信大学 MMA 部室
製本所　電気通信大学 MMA 部室

電気通信大学