出題日：7/20

ソフトウェア工学課題

第 13 回課題

学籍番号：33114098　野上恵吾

1　Sphinx で HTML ドキュメントを作成

今回ドキュメントを書くにあたり使用したプログラムは、Microsoft 製の Playwright というブラウザ自動化ツールを使用したスクレイピングプログラムである。これは Python で書かれており、今回はこれにドキュメントを追加した。なお今回使用したプログラムはすべて GitHub に上げているため、本レポートではそのリポジトリを参照する。

Playwright を使用した Python のソースコードが main.py と lib.py だ。lib.py を抜粋したものが Listing 1 だ。このソースコードのようにコメントをつけている。

そしてこれを Sphinx を用いて HTML ドキュメントを作成した。それが図 1 と 2 だ。この図より、Sphinx によるドキュメント生成がうまく働いていることが確認できる。

```
1  def number_of_h2_element(page):
2      """
3      h2要素の数を取得する関数
4
5      Args:
6          page(Page): context.new_page()した結果
7
8      Returns:
9          int: h2要素の数
10     """
11     return len(page.query_selector_all("h2"))
12
```

Listing 1: lib.py

図 1: lib.py の HTML ドキュメント

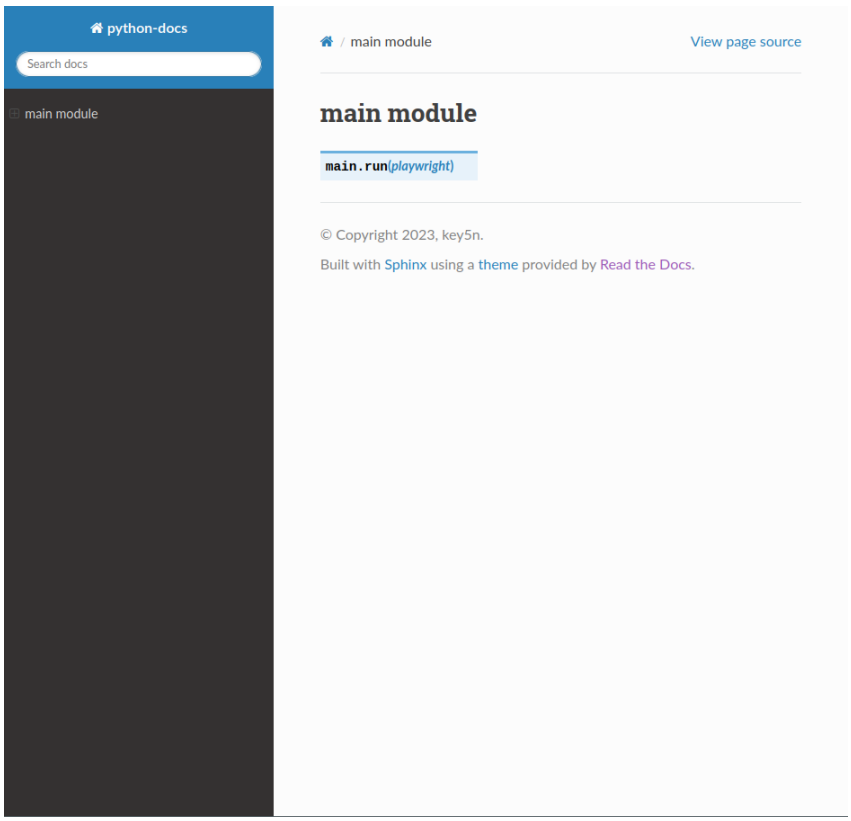

図 2: main.py の HTML ドキュメント

2　文書を Markdown で作成

次のことを行う。

1. Markdown ファイルを作成
2. それを pandoc で Word ファイルに変換
3. さらにそれを PDF に変換

まず Markdown ファイルを作成する。それが README.md だ。これにはこの課題で使用した Docker コンテナの所作や Playwright について軽く説明してある。次に pandoc で Word ファイルに変換した。その Word ファイルが図 3 だ。これには次のコマンドを使用した。

```
pandoc -f markdown -t docx README.md -o md_to_docx.docx
```

そして次に Word ファイルを pdf に変換する。この変換をするにあたり、まず pdf engine(lualatex) のインストールを行った。そして次のように実行した。

```
pandoc -f docx -t pdf md_to_docx.docx -o docx_to_pdf.pdf --pdf-engine=lualatex
```

その結果が図 4 である。これを見ると、日本語が出力されていないことがわかる。そのためフォント設定の header.tex を記述した。そして次のようにヘッダーファイルを読み込むような設定をして実行した。

```
pandoc -f docx -t pdf md_to_docx.docx -o docx_to_pdf.pdf --pdf-engine=lualatex --include-in-header=header.tex
```

これで生成された pdf ファイルが図 5 である。日本語が出力されていることが確認できる。

Get Started

コンテナのビルドをする

```
docker build -t scraping .
```

コンテナで python ソースコードを実行する

```
docker run -it --rm scraping
```

What's this

これは Microsoft が開発したブラウザ自動化ツールの **Playwright** を使用したプログラムです。 Playwright は主にフロントエンドの E2E テストやスクレイピングに使われます。 今回はタイトルや h1 要素のテキストの取得やスクリーンショットのみを実装しています。

Playwright 開発環境構築の仕方

1. Playwright パッケージをインストールする
2. Playwright からブラウザをインストールする（Playwright はそのときのブラウザの最新版を使用するため）

- 以下のブラウザを選べる
  - Chromium
  - Firefox
  - Webkit
  - Google Chrome
  - Microsoft Edge
  - Mobile Chrome
  - Mobile Safari

1. Playwright の System-dependencies をインストールする

図 3: README を Word ファイルに変換したもの

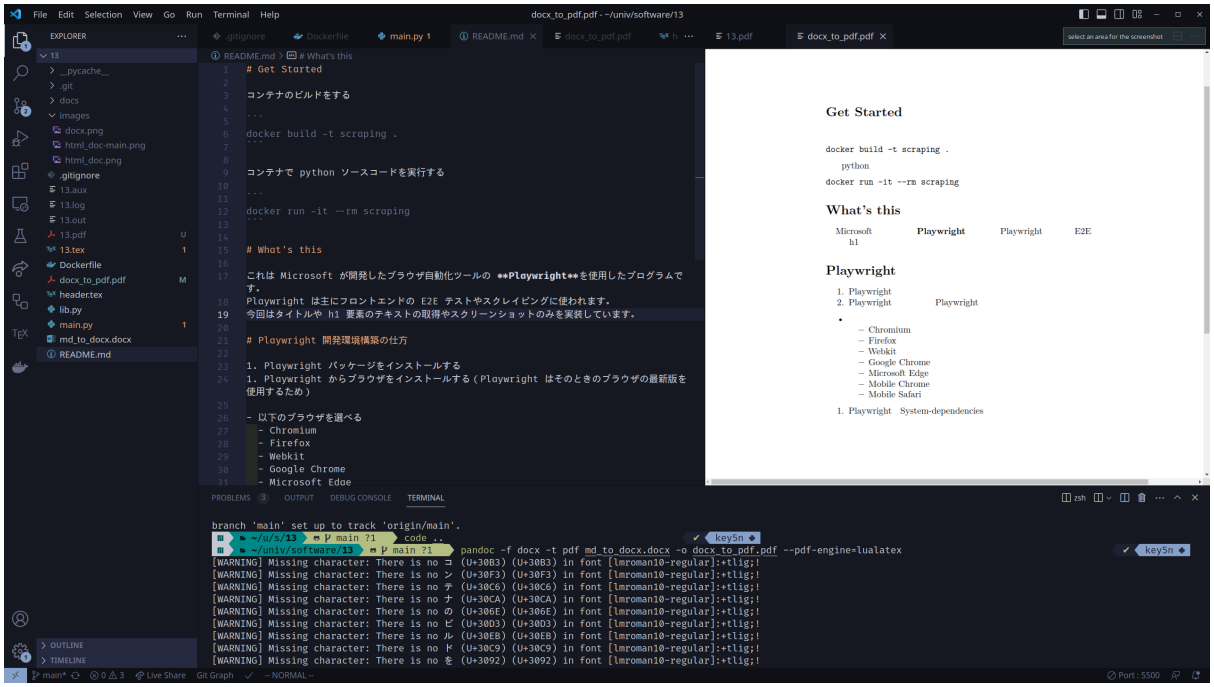

図 4: Word ファイルを pdf に変換したもの（失敗）

Get Started

コンテナのビルドをする

```
docker build -t scraping .
```

コンテナで python ソースコードを実行する

```
docker run -it --rm scraping
```

What’s this

これは Microsoft が開発したブラウザ自動化ツールの **Playwright** を使用したプログラムです。Playwright は主にフロントエンドの E2E テストやスクレイピングに使われます。今回はタイトルや h1 要素のテキストの取得やスクリーンショットのみを実装しています。

Playwright 開発環境構築の仕方

1. Playwright パッケージをインストールする
2. Playwright からブラウザをインストールする（Playwright はそのときのブラウザの最新版を使用するため）

- 以下のブラウザを選べる
  - Chromium
  - Firefox
  - Webkit
  - Google Chrome
  - Microsoft Edge
  - Mobile Chrome
  - Mobile Safari

1. Playwright の System-dependencies をインストールする

図 5: Word ファイルを pdf に変換したもの（成功）