Marpでスライド作成、カスタムテーマとGitHub Actionsによる自動化

Marpとは

https://marp.app/

> Marp (also known as the Markdown Presentation Ecosystem) provides an intuitive experience for creating beautiful slide decks. You only have to focus on writing your story in a Markdown document.

シンプルなMarkdown形式のテキストファイルからスライドを作成できます。
このスライドのソースはこちら。
https://github.com/75py/slide/blob/main/src/md/marp.md

marp-cli

https://github.com/marp-team/marp-cli

インストール方法はREADMEの通り。
このプロジェクトではローカルインストールとした。

```
npm install --save-dev @marp-team/marp-cli
```

```
{
  "name": "slide",
  "version": "1.0.0",
  "scripts": {
  },
  "devDependencies": {
+     "@marp-team/marp-cli": "^3.0.0"
  }
}
```

marp-cliの使い方

`marp` でヘルプが表示できる。今回使ったのは以下のとおり。

プレビュー表示

```
marp src/md/marp.md --theme src/theme/slide.css --preview
```

PDF出力

```
marp src/md/marp.md --theme src/theme/slide.css --pdf
```

npm runでテーマ指定を省略

```
{
  "name": "slide",
  "version": "1.0.0",
  "scripts": {
+     "preview": "marp $npm_config_src --theme src/theme/slide.css --preview",
+     "pdf": "marp $npm_config_src --theme src/theme/slide.css --pdf"
  },
  "devDependencies": {
    "@marp-team/marp-cli": "^3.0.0"
  }
}
```

使い方

- `npm run preview --src src/md/marp.md`
- `npm run pdf --src src/md/marp.md`

marp-cli オプション設定

> Marp CLI can be configured options with file, such as marp.config.js, marp.config.cjs, .marprc (JSON / YAML), and marp section of package.json. It is useful to configure settings for the whole of project.

例えば、テーマを固定したいだけなら `.marprc.yml` に以下を書くだけで実現可能。

```
theme: src/theme/slide.css
```

地味なはまりポイント：順序なし箇条書き

Markdownではアスタリスク、ハイフンで順序なし箇条書きを表現できるが、どちらを使うかでMarpの出力が変わる。

- ハイフン：一度に描画される
- アスタリスク：アニメーション描画される（次へ進むと表示される）

なお、当然ながらPDF出力では関係ない。

カスタムテーマ

built-in themes

default、gaia、uncoverの3種類がある。
どれもかなり見やすいテーマなので、色指定だけで済むのであればカスタマイズなしで利用するのも選択肢に入るように思う。

https://github.com/marp-team/marp-core/tree/main/themes

defaultを拡張したテーマを作成する

拡張する場合はこれだけで済む。

```
@import 'default';
```

WebStormだとdefaultが解決できずに赤線が引かれるが、無視して良い。
どうしても邪魔なら警告を消すことも可能。

```
+ /*noinspection CssUnknownTarget*/
@import 'default';
```

カスタマイズ

公式ドキュメント

https://marpit.marp.app/theme-css

たとえば、背景色を変えたいならsectionに指定すればいい。

```
section {
    background-color: lightblue;
}
```

企業なら会社ロゴを入れたりするといい感じ。

特定のページのみカスタマイズ

Markdownに以下を追加する。

```
<!-- _class: title -->
```

すると、出力されるHTMLは `<section class="title">` に変わるので、section.titleにCSS定義を追加すれば良い。

```
section {
    background-color: red;
}
```

GitHub Actionsによる自動化

今回想定する使い方

- mainブランチにマージされたら、MarkdownファイルからPDFファイルを作成する
- 作成されたPDFファイルをGoogleドライブに置いて共有する（artifactsで保存し、手動でアップロードする）
- GitHub Pagesで公開する

marp-cli-action を利用するのが一番簡単そう。
https://github.com/KoharaKazuya/marp-cli-action/blob/main/README.ja.md

ワークフロー（全文）

```yaml
name: Convert Markdown into PDF
on:
  push:
    branches:
      - main
  workflow_dispatch:

jobs:
  publish:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3

      - name: Convert Markdown into PDF
        uses: KoharaKazuya/marp-cli-action@v2
        with:
          config-file: ./.marprc-ci.yml
          generate-html: true
          generate-pdf: true

      - name: Save outputs
        uses: actions/upload-artifact@v2
        with:
          name: output
          path: ./output

      - name: Deploy to GitHub Pages
        uses: peaceiris/actions-gh-pages@v3
        with:
          github_token: ${{ secrets.GITHUB_TOKEN }}
          publish_dir: ./output
```

トリガー

```yaml
name: Convert Markdown into PDF
on:
  push:
    branches:
      - main
  workflow_dispatch:
```

- mainブランチがプッシュされたとき
- 手動実行

marp-cli-action

```yaml
jobs:
  publish:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3

      - name: Convert Markdown into PDF
        uses: KoharaKazuya/marp-cli-action@v2
        with:
          config-file: ./.marprc-ci.yml
          generate-html: true
          generate-pdf: true
```

html, pdfファイルを作成する。設定ファイルは `.marprc-ci.yml` を使用。

成果物を保存

```
      - name: Save outputs
        uses: actions/upload-artifact@v2
        with:
          name: output
          path: ./output
```

output.zipをダウンロードできるようになる。

出力されたファイルをGoogleドライブ等で共有したい場合はこちらを使う。

GitHub Pagesで公開する

```
      - name: Deploy to GitHub Pages
        uses: peaceiris/actions-gh-pages@v3
        with:
          github_token: ${{ secrets.GITHUB_TOKEN }}
          publish_dir: ./output
```

outputディレクトリのファイルを公開する。

GitHubの `Settings > Actions > General > Workflow permissions` を `Read and write permissions` にする必要あり。

成功すると、outputディレクトリ配下のファイルが gh-pages ブランチにプッシュされる。
https://github.com/75py/slide/tree/gh-pages

良い感じに設定すると、以下のようにアクセスできる。
https://75py.github.io/slide/marp.html

ローカルでのテスト実行

nektos/act を使う。
https://github.com/nektos/act

インストールは `brew install act` だけでOK。
もしDockerをインストールしていない場合はDockerも必要。

`act` で実行できる。