

技術ノート

---

# marp-templatesの解説

- テンプレート作成にあたってdocker化した理由
- Dockerのビルド手順
- スライド生成の手順
- デフォルトのmarkdownから拡張された書式の説明
- 各templatesの説明

# 1. Docker部分の説明

付箋はヘッダー設定部分に記述

# Docker化した理由

marpで下記を実現したいため、Docker化しました。

- ソースの**markdown**ファイルを複数に分割したい。(自動で結合したい)
- その際、特定の出力形式でのみ含めたい**ファイルの取捨選択を簡単に**したい。
- **markdownの拡張プラグイン**を組み込みたい。  
(そして、それを都度都度指定したくない)
- 拡張プラグインにもない**オリジナル書式をフィルター的に処理**したい。  
(自分でプラグイン化して公開というアプローチもありますが、今回は非対応)
- **OSによらず同一の動作**をするようにしたい。  
(スクリプトの使い分けなどしたくない)
- **上記全部盛り込もうと思ったらDockerにするしかなかった**

# Dockerのビルド手順

## 手順

一般的な[Dockerの手順](#)と同様です。スクリプト化しています。

1. [Githubからダウンロード](#)(clone)
2. [build-marp.sh|batを実行](#)

この手順で、markdownファイルの分割＆結合、  
markdownの書式の拡張、独自書式対応のフィルタ  
ー、の全てが組み込まれています。

## 補足

なぜ、marpをフォルダ分けしているかの理由ですが、  
marpと並行して使いたい他のツールもあるためで  
す。

つまり、将来の拡張のために事前に分けてます。

## 2. スライド生成

# スライド生成の手順

直接 `docker run` もできますが簡便にするためスクリプトを用意しています。  
各テンプレートフォルダ内に `slide-make.sh|bat` という名前で保存しています。  
以下、フォルダ構成と使い方です。

- 入力:  
markdownファイルは `src` フォルダに格納します。
  - `slide-make` 実行時に自動で結合されます。(拡張子 `md` が対象、名前順)
  - `css` や `img` など参照ファイルは `dist` フォルダに格納しておきます。
- 出力:  
生成されたスライドは `dist` フォルダに保存されます。
  - デフォルトファイル名は `slide.html` です。
- 生成:  
ターミナルから `slide-make.sh|bat` を実行すればOKです。
- 備考:  
`templates-01.minimum` で実行するとこのスライドが生成されます。

実行時のオプションとその内容は次ページ以降で説明します。

# スライド生成のオプション

オプションと機能を列挙します。動作は実際に実行して確かめてください。

- 基本(HTML生成、自動結合): `slide-make.sh`
- 結合時、一部ファイルを除外  
(ファイル名に指定文字列を含むかで判定)
  - `pdfonly` を含むファイルを除外:  
`slide-make.sh --exclude=pdfonly`
  - 複数指定はカンマ区切り:  
例) `slide-make.sh --exclude=pdfonly,htmlonly`
  - デフォルトで `-exclude` が組み込まれている:  
例) `作成用メモ-exclude.md`
- PDF生成: `slide-make.sh --pdf`
- PDF生成(`htmlonly`を除外):  
`slide-make.sh --pdf --exclude=htmlonly`
- テーマ設定(CSS指定):  
`slide-make.sh --css=<cssファイル名>`
  - デフォルトは `./css/style.css` (普段はこのまま)

- フィルター処理の入れ替え:

`slide-make.sh --filter=<filter名>`

- デフォルトはDocker内の `filter4mapr.py`
- filterはPythonスクリプトのみ。配置場所は `src` 内。
- 標準入力で結合mdを受け取り。標準出力でフィルター後mdを出力

以下はデバッグ用のオプションです。普通は使わないですが一応。

- markdownの結合をしない。marpによる変換のみ実行する
  - `slide-make.sh --convert`
  - 結合後のファイルは `dist/_merged.md` に保存されます。  
これを手動修正して確認したい場合のオプションです。
- デバッグモード(環境変数を表示。marpをデバッグモードで実行)
  - `slide-make.sh --debug`

### 3. 書式のお話

# markdown書式

markdownの書式について、拡張部分をメインに説明します。記述はサンプル的です。

md記述	変換後	備考
<code># text{.name}</code>	<code>&lt;h1 class="name"&gt;text&lt;/h1&gt;</code>	手軽にクラス付与できる
<code>[text]{.name}</code>	<code>&lt;span class="name"&gt;text&lt;/span&gt;</code>	手軽にspanできる
<code>::: name ~ :::</code>	<code>&lt;div class="name"&gt;~&lt;/div&gt;</code>	手軽にdivできる←手軽でない
<code>{{{name ~ }}}}</code>	<code>&lt;div class="name"&gt;~&lt;/div&gt;</code>	手軽にdivできる(独自フィルタ)
<code>\$e^{i\pi}+1=0\$</code>	$e^{i\pi} + 1 = 0$	数式表示(デフォ機能)

意味	タグ	記述	表示例	備考
強調1	mark	<code>=text=</code>	<code>text</code>	強調表示はこれが主流
強調2	em	<code>*text*</code>	<code>text</code>	デフォルト機能。非推奨
強調3	strong	<code>**text**</code>	<code>text</code>	デフォルト機能。非推奨
取り消し線	s	<code>~text~</code>	<code>text</code>	デフォルト機能
下線	u	<code>_text_</code>	<code>text</code>	-
下付	sub	<code>text~下~</code>	<code>text下</code>	-
上付	sup	<code>text^上^</code>	<code>text上</code>	-
ルビ	ruby	<code>{漢字 かんじ}</code>	<code>かんじ 漢字</code>	-

☞ callout対応  
こういうcallout的なものも変換できます

⚠ callout対応  
種類も色々です

# 除外ファイル実験

--exclude=htmlonly や --exclude=pdfonly の動作確認用の実験場です。  
--exclude オプションを指定しない場合、両方表示されます。

以下、オプションによる表示非表示場所：

※オプションに --exclude=pdfonly を指定するとこのメッセージは消えます。

# おまけ

mermaidにも対応できます。ただしHTMLオンリーです。(要JavaScript)

```
sequenceDiagram
    participant U as ユーザー
    participant A as アプリケーション
    participant D as データベース
```

```
U->>A: ログイン要求
A->>D: ユーザー情報照会
D->>A: ユーザー情報返却
A->>U: ログイン結果
```

Note over U,D: 認証フロー完了

# おしまい

以上で終了です。

簡単な資料ならこのテンプレートでsrc内のファイルを編集すれば作成できます。

しかし、見栄え良くするにはCSSの作り込みや、付随知識が必要になってきます。

その辺は気が向いたらテンプレートを追加していきます。

*fin*