Markdown の問題点: Kramdown と XeLaTeX

私の Jekyll ブログで Markdown を使って PDF を生成するために、以下の Pandoc コマンドを使用しています:

```
command = [
    'pandoc',
    input_markdown_path,
    '-o', output_pdf_path,
    '-f', 'markdown',
    '--pdf-engine', 'xelatex',
    '--resource-path=.:assets',
    '-V', f'CJKmainfont={CJK_FONT}',
    '-V', f'CJKsansfont={CJK_FONT}',
    '-V', f'CJKmonofont={CJK_FONT}',
    '-V', f'geometry:{GEOMETRY}',
    '-V', 'classoption=16pt',
    '-V', 'CJKoptions=Scale=1.1',
    '-V', 'linestretch=1.5'
]
```

Kramdown と XeLaTeX の両方をサポート

kramdown（Jekyll の HTML 出力用）と XeLaTeX（Pandoc 経由の PDF 出力用）の両方で動作する Markdown を書く際には、いくつかの考慮点があります:

1. 画像パスの互換性 • Kramdown (HTML): アセットを参照する際に、パスが / で始まることを好みます。 • XeLaTeX (PDF): 先頭に / のない相対パスを好みます。

解決策: 両方で動作する相対パスを使用する:

```
![](assets/images/chatgpt/block.jpg)
```

2. kramdown 属性の扱い • {:.responsive} は、HTML 出力のスタイリングに特化した kramdown の機能です。 • XeLaTeX はこれらの属性をサポートしておらず、エラーが発生します。

解決策: PDF 生成用の Markdown から kramdown 固有の属性を削除します。例えば:

```
![](assets/images/chatgpt/block.jpg){: .responsive }

![](assets/images/chatgpt/block.jpg)
```

もし `{:.responsive}` があなたの Jekyll HTML レイアウトにとって重要である場合、ウェブ出力には選択的に追加しつつ、PDF 生成プロセスではそれを省略することを検討してください。

デュアル互換性のためのワークフロー

1. kramdown 固有の機能への依存を最小限に抑えて Markdown コンテンツを作成します。
2. HTML での高度なスタイリングには、Markdown 内でインラインで適用するのではなく、Jekyll テンプレート内で直接 CSS クラスを適用します。
3. Markdown の移植性を維持しながら、Pandoc オプションを使用して PDF のフォーマットを制御します。

これらのプラクティスに従うことで、Markdown コンテンツは Jekyll の HTML レンダリングと XeLaTeX による PDF 生成の両方で互換性を保ち、マルチフォーマット出版のためのシームレスなワークフローを確保します。