Re:VIEW+CSS組版 環境構築

～グランパとマゴと一緒に環境構築～

@at_grandpa

はじめに

本書の特徴

この本は、Re:VIEW+CSS組版の環境構築についての書いたものです。開発環境のサンプルPDFも兼ねています。Re:VIEW記法については本書の範囲を越えるので、本家のドキュメントを御覧ください。

環境構築のリポジトリは以下です。

- https://github.com/at-grandpa/review-and-css-typesetting

このリポジトリで生成できるものがこのPDFになります。説明どおりに実行すれば生成できますので、ぜひ一度お試しください。

フィードバック

PRは大歓迎です。よりよいRe:VIEW+CSS組版環境を作っていきましょう！下記からissueを立ていただき、`@at_grandpa`までご連絡ください。

- https://github.com/at-grandpa/review-and-css-typesetting/issues/new

お待ちしています。

目次

第1章　環境構築と執筆の流れ

さて、Re:VIEW+CSS組版の環境構築について説明するぞい。

わ！こんなところに私たちも登場していいの！？

いいんじゃよ！張り切って説明するぞい！

こんにちは。@at_grandpaです。今回は、Re:VIEW+CSS組版の開発環境についてお話します。マゴとグランパと一緒に進めていきましょう。さっそく使い方を見ていきます。

1.1　動作環境

まずは動作環境です。次の動作環境のみで確認をしています。もし他の環境で問題が起こる場合は、自前で修正していただくか、issueを送っていただけますと幸いです。

```
$ sw_vers | grep Product
ProductName:    Mac OS X
ProductVersion: 10.13.6

$ docker -v
Docker version 18.09.1, build 4c52b90

$ docker-compose -v
docker-compose version 1.23.2, build 1110ad01

$ make -v
GNU Make 3.81
```

1.2 git clone

ではリポジトリのcloneを行いましょう。次のコマンドをたたき、リポジトリのルートディレクトリへ移動します。

```
$ git clone https://github.com/at-grandpa/review-and-css-
typesetting.git
$ cd review-and-css-typesetting
```

個々人のリポジトリで使用する場合、cloneしたディレクトリ群をコピーして使用してください。

1.3 執筆するファイルの配置場所

執筆するファイルの配置場所は `./articles` 以下です。このディレクトリ内は、Re:VIEWのファイル配置規則にしたがってください。`review-init` で生成されるディレクトリを `./articles` に置き換えていただければ大丈夫です。もし最初から執筆を行う場合は、このあとに説明するコマンド `make init` を叩いてください。このコマンドは、`./articles` ディレクトリ内を初期化（`review-init articles`を実行した状態）するコマンドです。

1.4 コマンドの使い方

主なコマンドを見ていきましょう。次のコマンドはすべてリポジトリルートで実行するコマンドです。

- `make setup`
  - Re:VIEW+CSS組版環境の構築を行う
- `make init`
  - `./articles` 内を初期化する
- `make pdf`
  - PDFを生成する
- `make browser`
  - vivliostyleを用いてブラウザで開く
  - デベロッパーール等でCSSの確認ができる

- `make lint`
    - textlint, prh によるlint
- `make lint/fix`
    - prh の自動修正

リポジトリのclone後は `make setup` を叩いて環境を構築します。環境を構築したら、 `make pdf`, `make browser`, `make lint` などを使用して執筆していきます。

その他のコマンドについてはhelpを参照してください。`make` とたたくか `make help` とたたくことでhelpを表示できます。

```
$ make help

    review-and-css-typesetting

    usage: make [command]

    help        ヘルプを表示
    args        デフォルトの変数を表示する
    setup       docker環境をセットアップする
    build       dockerイメージをbuildする
    up          dockerコンテナを立ち上げる
    clean       dockerコンテナを停止して削除する
    stop        dockerコンテナを停止する
    rm          dockerコンテナを削除する
    ps          dockerコンテナー覧を表示する
    login       dockerコンテナ内にログインする
    init        ./article配下を削除して、review-initで./articlesを再生
成する
    html        htmlを生成する (./articles/book.html)
    pdf         PDFを生成する (default: ./articles/book.pdf) (出力先変
更: make pdf PDF=./hoge.pdf)
    browser     ブラウザで表示する (vivliostyle経由)
    lint        textlint, prhでlintをかける
    lint/fix    prhの指摘点を自動修正する
```

1.5 執筆の流れ

続いて、具体的な執筆の流れを説明していきます。

1.5.1 *.re ファイルを書いていく

Re:VIEW記法で書く *.re ファイルは ./articles ディレクトリに配置します。このディレクトリの中に catalog.yml があるので、それに沿って書いていきます。詳しくは、Re:VIEW本家のドキュメントを御覧ください。

1.5.2 ブラウザで内容を確認する

ある程度執筆したら、ブラウザで内容を確認しましょう。make browser コマンドをたたくとブラウザで内容を確認できます（openコマンドでURLを叩いています。macOS限定です。環境によってMakefileを書き換えてください）。vivliostyle経由で閲覧でき、Chrome DevTools などを用いればCSSの一時的な変更もできます。微調整に最適です。

1.5.3 CSSを編集する

CSSファイルは ./articles/style.css です。このPDF生成の style.css は、vvakameさんのCSS組版リポジトリのファイルを参考にしています。対象のhtmlは ./articles/book.html です。ブラウザでの確認をもとに、CSSファイルを編集していきましょう。見た目の確認は make browser でもできますが、最終的にできあがるPDFとは若干デザインが異なるのでPDFの確認も行いましょう。

1.5.4 lintをかける

make lint でtextlintとprhを用いたlintをかけることができます。デフォルトでは ./textlintrc の設定を元にしています。preset-ja-technical-writing や period-in-list-item のルールはすでにインストール済みです。prhの設定に関しては、 ./prh-rules ディレクトリに配置しています。次のリポジトリを参考にさせていただきました。

- https://github.com/prh/rules

適宜必要な設定に書き換えて使用してください。

1.5.5　PDFを生成する

PDF生成は `make pdf` コマンドです。目次のリンクやURLのリンクも適用済みです。`make browser` の結果とは若干異なるので、最終的なチェックはPDFで行いましょう。

1.5.6　circleciでPDF生成やlintをチェックする

`./.circleci/config.yml` にcircleciでの設定が書かれています。lintとPDF生成を行っています。必要であれば適宜書き換えてください。

1.5.7　フォントの埋め込みをする

印刷所に入稿するときは、PDFにフォントを埋め込まなければいけません。自動で埋め込みができているとよいのですが現状できていません。macOSの場合は `プレビュー.app` を用いて再度PDFに書き出すことでフォント埋め込みが可能です。他の環境の方は、フォントの埋め込み方を調べて実施してください。

1.6　ちょっとした注意点

自分自身が陥った罠なのですが、「デザインに凝ると時間が無限に必要」です。まずは内容を完成させましょう。最悪、デザインはシンプルでも内容が充実していれば伝わります。逆に、デザインが良くても内容が不十分だと読者はがっかりします。この順番は間違えないようにしてください。

（自分にも言い聞かせよう。。。）

第2章　独自設定の説明

だいたいわかったかの？

うん！わかったー！だけど、デザインに凝りすぎるのもよく無いね。気をつけよー。

そうじゃな。あくまでも内容が一番大切じゃ。

次はなんの説明なの？

次は、このリポジトリの独自設定の説明じゃ。
Re:VIEWの拡張など、少し特殊なことをやっておる。その説明をするぞい。

はーい！

さて、次はグランパのいうとおり「このリポジトリの独自設定」を説明します。編集の参考にしてみてください。

2.1　大扉のデザイン

大扉のデザインは、自分のCSS力では難しかった（CSSだけで表現できなかった）ので、Rubyのライブラリ `Nokogiri` を用いてhtmlを編集しています。`in_docker.mk` の次の部分がhtml生成箇所です。

```
html:
        cd articles/ && \
                review-epubmaker config.yml && \
                review-epub2html book.epub | \
                bundle exec ruby ../scripts/html-ext.rb > book.html
```

- config.yml をもとにepubを生成
- review-epub2html でepubをひとつのhtmlに変換
- ./scripts/html-ext.rb を通してhtmlタグを修正
- htmlに出力

必要なhtml要素が準備できたら、後はCSSを編集するだけです。./articles/style.css も参考にしてみてください。

また、これらのデザインを決める際は make browser を用いてブラウザに表示し、CSSを直接いじって試行錯誤を繰り返しました。

2.2 目次のデザイン

これも大扉の場合と同じです。./scripts/html-ext.rb を用いて目次部分に必要なhtmlタグを追加し、CSSでデザインを適用しています。

2.3 見出しのデザイン

こちらは純粋にCSSだけで実現しています。生成される ./article/book.html の中身をみて、いろいろ試行錯誤しました。お好きな形に変更していただいてかまいません。

2.4 「コラム」や「問題」「答え」のデザイン

「コラム」や「問題」「答え」のデザインは、Re:VIEWの公式の拡張 ./articles/review-ext.rb を使っています。拡張に関しては、次の「Re:VIEWのモンキーパッチによる拡張の基本」と ./articles/review-ext.rb を参考にしてください。

- Re:VIEW のモンキーパッチによる拡張の基本
  - https://review-knowledge-ja.readthedocs.io/ja/latest/reviewext/review-ext-basic.html

拡張したコマンドによって生成されたhtmlに対し、CSSを適用しています。もちろん、色やマージンなどの微調整も可能です。

コラム

```
===[mycolumn] コラム

こんな感じでコラムを書いています。

===[/mycolumn]
```

見出しも書けます

見出しも書けていますね。

問題

これは問題です。次のように書いています。

```
===[question] 問題だよ

これは問題です。以下のように書いています。

===[/question]
```

キャプションの文字列は「問題だよ」のように任意の文字列を設定できます。

答えだよ

これは答えです。「問題」と同様、キャプションの文字列は自由に設定できます。

```
===[answer] 答えだよ

これは答えです。「問題」と同様、キャプションの文字列は自由に設定できます。

===[/answer]
```

2.5 吹き出しのデザイン

左側の吹き出しじゃ。

右側の吹き出しだよー！

長文にも対応しておるぞ。長文にも対応しておるぞ。長文にも対応しておるぞ。長文にも対応しておるぞ。長文にも対応しておるぞ。長文にも対応しておるぞ。長文にも対応しておるぞ。長文にも対応しておるぞ。長文にも対応しておるぞ。長文にも対応しておるぞ。長文にも対応しておるぞ。長文にも対応しておるぞ。

改行にも対応しているよ！
改行にも対応しているよ！
改行にも対応しているよ！
改行にも対応しているよ！
改行にも対応しているよ！

前回の技術書典にて頒布した本で、一番やりたかったことです。こちらもRe:VIEWの公式の拡張 `./articles/review-ext.rb` を使っています。次の独自コマンドを作成し、 `.re` ファイルで使用しています。

- `imagetalkl（image-talk-left）`
- `imagetalkr（image-talk-right）`

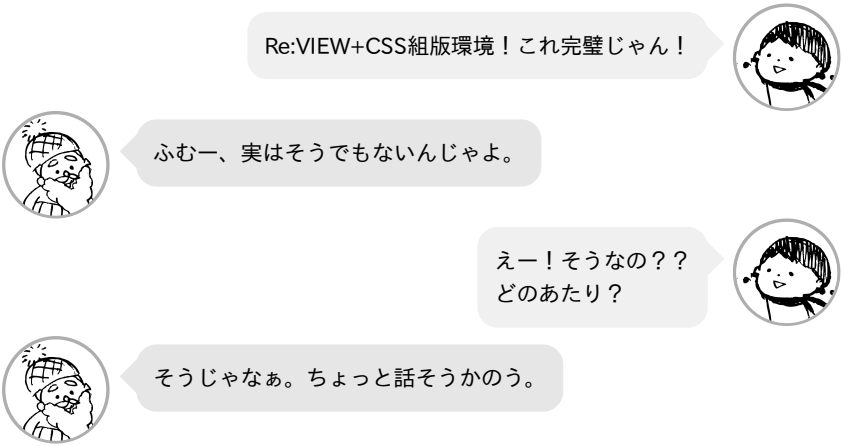
`*.re` ファイルで使用している箇所があるので探してみてください。

ひとつ注意点があります。RE:VIEWでは、同じ画像ファイルを複数回使うと「警告：画像IDが重複しています」と出力されます。

```
WARN: warning: duplicate ID: grandpa (#
<ReVIEW::Book::ImageIndex::Item:0x0055e62fb5fd18>)
WARN: warning: duplicate ID: grandpa (#
<ReVIEW::Book::ImageIndex::Item:0x0055e62f52ef20>)
WARN: warning: duplicate ID: mago (#
<ReVIEW::Book::ImageIndex::Item:0x0055e62f52edb8>)
WARN: warning: duplicate ID: grandpa (#
<ReVIEW::Book::ImageIndex::Item:0x0055e62f52ec28>)
WARN: warning: duplicate ID: mago (#
<ReVIEW::Book::ImageIndex::Item:0x0055e62f52ea98>)
```

最終生成物には影響はありませんでした。気になる方はPRをいただけると幸いです。

第3章　やりたかったこと



3.1　「柱」に章・節を表示したい

各ページの上部に、章や節の見出しの文字列が表示されます。これを「柱」と言います。現状では「Re:VIEW+CSS組版環境構築」と表示されています。これは`style.css`にハードコーディングされています。使い回す際は書き換えてください。

この柱には、章・節を表示したいです。CSSの機能で表示できるかと思っていましたが、なかなかうまくできませんでした。

3.2　CSSをいい感じにしたい

`./articles/style.css`にべた書きしています。CSS周りの知識が乏しいので、現状だとこのようになっています。もっと扱いやすくできるはずですので、適宜アップデートしていきます。

3.3　フォントを豊富にしたい

フォント周りの知識も乏しいので、自由に入れられる環境になっていません。現在はそれっぽいフォントを選択していますが、もっと自由度を上げていきたいです。

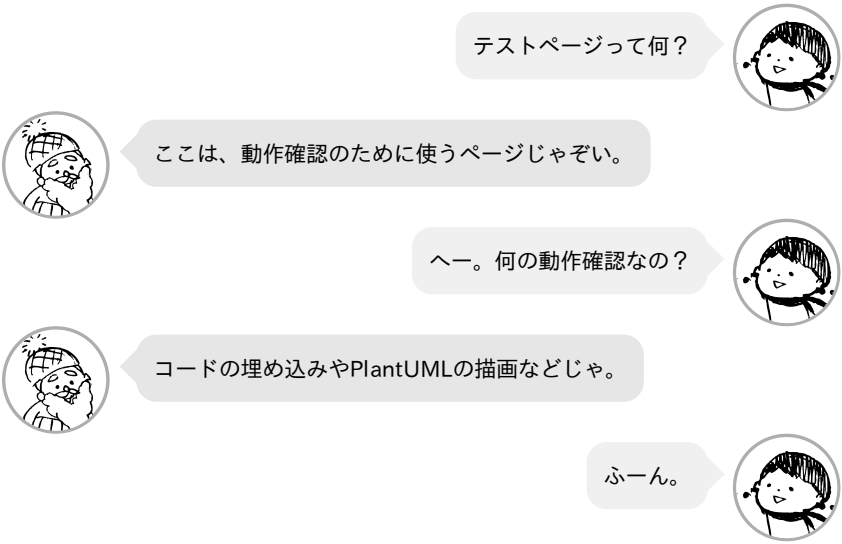
3.4　脚注をいい感じにしたい

脚注に関してはノータッチです。もしうまくできた方は、ご一報ください。

このように、やりたいことはまだまだたくさんあります。時間に余裕があった場合は、適宜アップデートしていく予定です。

（あまり期待はしないでください。。。）

第4章　テストページ



4.1　コードの埋め込み

Re:VIEWには外部ファイルを埋め込む機能があります。

▼コードの埋め込み

```
class Foo
  def display(str : String)
    puts str
  end
end

foo = Foo.new
foo.display("Hello!!") # => Hello!!
```

このコードは `./code/sample.cr` に書かれているものです。`review-preproc --replace *.re` のコマンドを実行し、 `*.re` 内に書かれている `@mapfile` 命令を置換します。

4.2 PlantUML

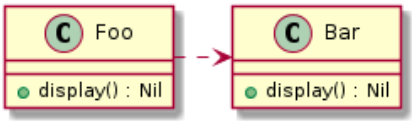
Re:VIEWは `glaph` 命令でPlantUMLを描画できます。次のコードを `glaph` 命令に書くことで、次のような図を埋め込むことができます。

▼リスト4.1: PlantUMLのコード

```
@startuml
Foo .r.> Bar

class Foo {
    + display() : Nil
}

class Bar {
    + display() : Nil
}
@enduml
```



▲ 図4.6: PlantUMLの描画

この他、できることが分かり次第、随時追加していきます。

あとがき

今回は「Re:VIEW+CSS組版環境」について書きました。「できるだけ簡単に環境を構築できる」を目指しました。前回の技術書典５ではこれと似た環境でやっていましたが、vivliostyleのサーバ立ち上げやPDFの出力に多少時間がかかっていました。その部分の短縮も多少できています。

しかし、まだまだCSS組版の機能を網羅できていません。ぜひPRなり修正ブログだったり、何かしらの形でフィードバックをいただけたら幸いです。

よりよい執筆環境を目指していきましょう！

Re:VIEW+CSS組版 環境構築
～グランパとマゴと一緒に環境構築～

2019年1月14日　v1.0.0

著　者	@at_grandpa