

Cloche

레이텍 사용자를 위한 ePub 변환기



머리말

A4 크기의 PDF를 스마트폰에서 보는 것이 미묘하게 불편하다. A4 크기와 더불어 A5나 더 작은 크기의 PDF를 동시에 만들어내는 것이 레이텍 사용자들에게 거의 전혀 수고스러운 일이 아니다. 그러나 내 스마트폰의 화면 크기에 알맞는, 여백 없는 페이지 크기를 정하는 것이 그다지 쉽지 않다. 그리고 기대와 달리 그 불편한 느낌이 페이지 크기에 비례하여 감소하지 않는다.

ePub은 HTML 파일들로 이루어지지만 HTML과 같지 않다. HTML로 문서를 만들 때 용납하기 어려운 두 가지 문제가 발생한다. 하나는 다수의 파일들이 하나의 문서를 구성한다는 것이다. 그 점이 문서의 관리와 배포를 어렵게 만든다. 다른 하나는 HTML 파일들을 조직화하기가 쉽지 않다는 것이다. 쉽게 말해 HTML 페이지에 다른 페이지들로 연결되는 링크들을 만들어줘야 한다. ePub에서 이 문제들이 사라진다.

솔직히 말하자면, 애플 도서 Apple Books 앱에서 페이지 넘김이 무시할 수 없을 만큼 매혹적이다. 그리고 텍스트 리플로우 text reflow가 노안 老眼에게 필수적인 기능이다.



웹 브라우저와 달리 ePub 리더들은 매스작스 MathJax를 지원하지 않는다. 수식을 이미지로 만들어 ePub에 삽입하는 것은 매우 고단한 작업이 될 것이다. 그 점을 제외하면 ePub이 PDF에 비해 취약하다고 말할 것이 없다.

어떻게 ePub을 만들 것인가? 시길 Sigil이 만족스러운 ePub 에디터라 말하지 못하겠다. 설령 그것이 제법 괜찮은 것이라고 해도 나의 선택지에 포함될 수 없다. 스텝스 Sphinx나 주피터

노트북 Jupyter Notebook 과 같은 부류도 고려 대상이 아니다. 내가 원한 것은 레이텍에서 ePub 을 만들어내는 것이다. 팬독 Pandoc 이 유일하게 나의 기대에 부합한다.

레이텍의 최고의 장점 중 하나가 사용자가 매크로들을 만들 수 있다는 것이다. 팬독이 경이로운 프로그램이지만, 나는 사용자 정의 매크로들을 팬독에게 이해시키는 방법을 알지 못한다. 단순한 방법을 시도하기로 했다. 정규 표현식을 이용하여 레이텍 매크로들을 HTML 태그로 바꾸는 것이다. 기대 이상으로 만족스러웠던 그 소박한 시도의 결과가 개인 프로젝트로서 계속하도록 나를 고무하였다.

Cloche 클로시 는 레이텍 latex 파일을 ePub으로 변환하는 파이썬 스크립트이다. 처음에 trans-
tex이라 이름지었으나, 그 이름은 용도를 드러내지도 못하고 근사하지도 않다.

클로시는 주로 여성들이 착용하는 종 모양의 모자이다. 소설가 이 민진이 『파친코』를 발표하기 전에 펴낸 『백만장자를 위한 공짜 음식』에서 클로시를 배웠다. 그 소설에서 주인공 케이시 한이 톱툰이 모자를 만들어 판다. 세상에 매우 멋진 모자들이 아주 많다. 클로시는 우아하다. 나의 클로시 스크립트도 그렇게 우아해지기를 희망한다.

클로시 스트립트가 우아하지 않은 까닭은 사용자에게 정규 표현식을 요구하기 때문이다. 유감이지만, 정규 표현식을 모른다면 클로시의 사용을 포기해야 한다.

차례

— 1 —

개요

Cloche는 설정 파일을 요구한다.

```
C:\>cloche.py [config.yml]
```

설정 파일이 지정되지 않으면 `_html/epub.yml`이 참조된다.

1.1 주요 과정

클로시의 주요 처리 과정은—디폴트 설정을 기준으로—다음과 같다.

1. `create_epub_directory()` 함수가 `_epub/` 디렉토리를 만든다. 그다음에 `create_basic()` 함수가 `_epub/mimetype` 파일과 `_epub/META-INF/container.xml` 파일을 만든다.
2. `copy_additional()` 함수가 `additional` 설정 옵션에 지정된, `cover.xhtml`이나 `basic.css` 같은 파일들을 `_epub/OEBPS/` 디렉토리로 복사한다.
3. `copy_tex()` 함수가 `tex_files` 설정 옵션에 명시된 파일들을 —확장자를 `.xhtml`로 바꾸어— `_epub/OEBPS/` 디렉토리로 복사한다. 이때 `escape` 설정 옵션에 지정된 `.tsv` 파일이 있다면, `wordig` 모듈이 그 파일의 지시에 따라 특정 문자열들을 다른 것들로 바꾼다. ?? 페이지 ??를 보라.
4. `add_heading_id_tex()` 함수가 —또는 `add_heading_id_html()` 함수가— 장절 명령들을 HTML 태그로 치환한다. ?? 페이지 ??와 ?? 페이지 ??를 보라.
5. `create_endnote()` 함수가 각주들을 모아 미주 페이지를 만든다. ?? 페이지 ??를 보라.
6. `create_index()` 함수가 색인 페이지를 만든다. ?? 페이지 ??를 보라.
7. `converter` 설정 옵션에 지정된 하나 이상의 `.tsv` 파일들을 참조하여 `wordig` 모듈이 모든 레이텍 매크로들을 HTML 태그들로 치환한다. ?? 페이지 ??를 보라.
8. `copy_images()` 함수가 `original_image_directory` 설정 옵션에 지정된 디렉토리에서 실제 문서에 삽입되는 이미지 파일들을 `_epub/OEBPS/images/` 디렉토리로 복사한다. ?? 페이지 ??를 보라.
9. `create_ncx()` 함수가 `toc.ncx` 파일을 만든다. ?? 페이지 ??를 보라.
10. `create_opf()` 함수가 `content.opf` 파일을 만든다. ?? 페이지 ??를 보라.

11. `zip_epub()` 함수가 `_epub/` 디렉토리를 압축하여 `.epub` 파일을 만든다.

1.2 스크립트 옵션

Cloche는 ePub을 새로 만들 때마다 `_epub/` 디렉토리를 비운다.

```
C:\>cloche.py -k [config.yml]
```

이전 파일들을 남겨두려면 `-k` 옵션을 사용하라.

1.3 자매 스크립트

Cloche가 정상적으로 작동하려면 아래 스크립트들이 함께 있어야 한다.

`wordig.py` 이 스크립트가 아주 다양한 기능들을 제공하지만, Cloche를 위해서는 오로지 문자열 찾기-바꾸기를 수행한다.

`josa.py` 이것이 \을 같은 자동조사 매크로들을 처리한다. ?? 페이지 ??를 보라.

`hanja2hangul.py` 이것은 한자 발음 사전이다. `josa.py`가 사용한다.

`op.py` 이것은 Cloche와 직접적인 관련이 없으나 `wordig.py`가 요구하기 때문에 필요하다.

`docenv.conf` 이것은 `op.py`가 참조하는 설정 파일이다. 없어도 무방하다.

이 문서의 PDF를 만드는 데에 HzGuide 클래스가 사용되었다. HzGuide 클래스 설명서의 부록에서 위 스크립트들에 대한 간략한 설명이 제공된다. HzGuide 클래스와 그 설명서를 <https://github.com/YiHoze/HzGuide>에서 얻을 수 있다.

— 2 —

ePub 구조

하나의 epub 파일에는 다음과 같은 구조의 디렉토리가 압축되어 있다.

```
\_epub\  
  +---OEBPS\  
    +---images\  
      +---image_1.jpg  
      +---image_n.jpg  
    +---fonts\  
      +---font.otf  
    +---chapter_1.xhtml  
    +---chapter_n.xhtml  
    +---style.css  
    +---content.opf  
    +---toc.ncx  
  +---META-INF\  
    +---container.xml  
  +---mimetype
```

mimetype 파일에 미디어 타입이 다음과 같이 선언되어야 한다.

```
application/epub+zip
```

container.xml이 다음과 같이 작성되어야 한다.

```
<?xml version="1.0" encoding="UTF-8"?>  
<container version="1.0"  
  ↪ xmlns="urn:oasis:names:tc:opendocument:xmlns:container">  
  <rootfiles>  
    <rootfile full-path="OEBPS/content.opf"  
      ↪ media-type="application/oebps-package+xml"/>  
  </rootfiles>  
</container>
```

content.opf는 이미지를 비롯하여 ePub에 포함되는 모든 파일들의 명세서이다. 아울러 문서

제목 같은 메타데이터가 포함된다.

```
<dc:title>Cloche</dc:title>
<item id="ncx" href="toc.ncx" media-type="application/x-dtbncx+xml" />
<item id="cover.xhtml" href="cover.xhtml" media-type="application/xhtml+xml"/>
<item id="basic.css" href="basic.css" media-type="text/css"/>
<item id="foo.xhtml" href="foo.xhtml" media-type="application/xhtml+xml"/>
```

ePub 파일에 잉여 파일들이 포함되어 있는 것은 문제를 일으키지 않지만, content.opf에 기재된 파일이 누락되어 있다면 그 ePub 파일은 손상된 것으로 간주된다.

toc.ncx가 PDF 책갈피와 같은 역할을 한다.

```
<navPoint id="navPoint-1" playOrder="1">
  <navLabel><text>CHAPTER TITLE</text></navLabel>
  <content src="foo.xhtml#heading_1"/>
  <navPoint id="navPoint-2" playOrder="2">
    <navLabel><text>SECTION TITLE</text></navLabel>
    <content src="foo.xhtml#heading_2"/>
  </navPoint>
</navPoint>
```

— 3 —

설정

설정 파일이 YML 형식으로 작성되어야 한다. 달리 지정하지 않으면 클로시는 `_html/epub.yml` 파일을 읽으려고 한다.

```
output_directory: _epub
output_file: cloche
tex_files:
  - cloche.tex
additional:
  - type: html
    file: cover.xhtml
    from: _html
    to: OEBPS
  - type: css
    file: basic.css
    from: _html
    to: OEBPS
escape: _html/escape.tsv
indexer: _html/indexer.tsv
converter: _html/tex2xhtml.tsv
original_image_directory: images
metadata:
  title: "Cloche: 레이텍 사용자를 위한 ePub 변환기"
  creator: 이 호재
  publisher: 이 호재
  rights: © 2022 이 호재
  language: ko-KR
```

`output_directory` 출력 디렉토리를 지정하라. 이 옵션이 지정되지 않으면 `_epub`으로 설정된다.

`output_file` 출력 파일을 지정하라. 이 옵션이 지정되지 않으면 오늘 날짜가 파일 이름으로 사용된다.

`tex_files` 레이텍 파일들을 지정하라.

`html_files` HTML 파일들을 지정하라.

`additional` 추가할 파일들을 지정하라.

`escape` ?? 페이지 ??를 보라.

`converter` ?? 페이지 ??를 보라.

`original_image_directory` 이미지 파일들이 포함된 디렉토리를 지정하라. ?? 페이지 ??를 보라.

`metadata` 문서 제목과 저자 이름, 그리고 나머지 메타데이터를 명기하라.

3.1 소스 파일

`tex_files` 설정 옵션 아래에 포함할 레이텍 파일들을 지정하라.

```
tex_files:
- foo.tex
- goo.tex
- hoo.tex
```

HTML 파일을 포함하려면 `html_files` 설정 옵션을 사용하라.

```
html_files:
- foo.html
- goo.html
- hoo.html
```

텍 라이브에 포함된 `lwarpmk`를 이용하여 레이텍 파일에서 HTML을 만들 수 있다는 점이 고려되었다. 그밖에도 HTML 파일들을 만드는 여러 방법들이 있을 것이다. 그러나 레이텍 파일과 HTML 파일을 섞는 것은 허용되지 않는다. `tex_files` 옵션이 설정되어 있다면 `html_files` 옵션이 무시된다. 이것은 처리하기 번잡하기 때문이라기보다 그 두 형식을 동시에 사용해야 하는 경우를 상상하기 어렵기 때문이다.

소스 파일들이 `_epub/OEBPS/` 디렉토리로 복사될 때 확장자가 `.xhtml`로 바뀐다.

3.2 부수 파일

```
additional:
- file: cover.xhtml
  from: _html
  to: OEBPS
- file: basic.css
  from: _html
  to: OEBPS
```

이 설정에 의해 `_html/cover.xhtml` 파일과 `_html/basic.css` 파일이 `_epub/OEBPS/` 디렉토리로 복사된다.

폰트나 오디오 파일을 같은 방법으로 추가할 수 있다.

```
additional:
- file: NotoSerifKR-Regular.otf
  from: _html
  to: OEBPS/fonts
- file: BTS_Dynamite.mp3
  from: _html
  to: OEBPS/audio
```


— 4 —

마크업 변환기

wordig 모듈이 레이텍 매크로들을 HTML 태그들로 치환한다. 여기에는 TSV 형식으로 작성된 변환 명세 파일이 필요하다.

converter 옵션에 변환 명세 파일이 지정되어야 한다.

```
converter: _html/tex2xhtml.tsv
```

변환 명세 파일에서 찾기-바꾸기를 나타내는 한 쌍의 정규 표현식과 치환 문자열이 각 줄을 차지한다.

```
~~~ DOTALL
\\documentclass.*(?:\\begin\\{document\\})
~~~
\\begin\\{verbatim\\}\\n <pre>
\\end\\{verbatim\\} </pre>
\\textbf\\{(.+?)\\} → <b>\\1</b>
```

변환 명세 파일에 다음과 같은 규칙이 적용된다.

- 빈 줄이 무시된다.
- 둘째 칸이 없거나 비어있으면 대상 문자열이 삭제된다.
- “~~~”로 시작하는 줄이 주석으로 간주된다.
- 정규 표현식에서 도트(.)는 개행 문자를 제외한 모든 문자에 대응된다. 주석 줄 끝에 DOTALL이 있으면, 그 다음 주석 줄을 만날 때까지 그 아래에 있는 모든 정규 표현식들의 처리에서 도트가 개행 문자를 포함하는 모든 문자에 대응된다. 따라서, 위 예에서 \\documentclass부터 \\begin{document}까지 그 사이에 있는 모든 줄이 삭제된다.

참고

한 파일의 내용이 정규 표현식들에 의해 반복적으로 바뀌기 때문에 정규 표현식들이 나열되는 순서가 중요하다는 점을 유념하라.

조건부 텍스트 `conditional text`에 대응하도록 복수의 변환 명세 파일들을 지정할 수 있다.

```
converter:
- _html/tex2xhtml_adhoc.tsv
- _html/tex2xhtml_common.tsv
- C:/home/_html/tex2xhtml.tsv
```

여러 변환 명세 파일에 동일한 정규 표현식이 있을 때, 먼저 주어진 것에 의해 문자열이 치환되므로 결과적으로 뒤의 것은 무시된다.

4.1 XHTML 페이지의 처음과 마지막

XHTML 페이지는 `<?xml`로 시작해서 `</html>`로 끝나야 한다. 그것들을 삽입하기 위해, 이 문서에 사용된 HzGuide 레이텍 클래스에 정의된 바와 같이, 아무 것도 하지 않는 매크로가 필요하다.

```
\NewDocumentCommand\htmlbegin{}
\NewDocumentCommand\htmlend{}
```

변환되어야 할 레이텍 파일들이 `\htmlbegin`과 `\htmlend`를 포함해야 한다. 그리고 그것들에 대한 치환 지시가 변환 명세 파일에 포함되어야 한다.

```
\\htmlbegin → <?xml ...>...</body>
\\htmlend → </body>\n</html>
```

대부분의 경우에 하나의 ePub 파일이 여러 XHTML 파일들로 이루어진다.

```
\input{foo.tex} (chapter) ⇒ \input{foo_1.tex} (section)
                             \input{foo_2.tex} (subsection)
                             \input{foo_3.tex} (section)
```

이 경우에 다음과 같이 설정할 수 있다.

```
tex_files:
- foo.tex
- foo_1.tex
- foo_2.tex
- foo_3.tex
```

그러나 탐색 제어 파일 `toc.ncx`에서 기대와 다르게 절 제목들이 장 제목의 수준을 갖게 될 것이다. 이 문제를 피하기 위해 `copy_tex()` 함수가 레이텍 파일들을 목적 디렉토리로 복사할 때, `merge_tex()` 함수가 `\input` 명령이 내용에 포함되어 있는지 확인하고, 있으면 해당 줄을 대상 파일의 내용으로 치환한다. `\input` 명령은 일 회만 처리된다. 즉 치환하는 내용에 포함된 `\input` 명령들은 무시된다.

4.2 장절 제목

장절 명령들이 `add_heading_id_tex()` 함수에 의해 `\subsection`까지 다음과 같이 바뀐다.

```
\chapter{장 제목}\label{chp} ⇒ \chapterline{장 제목}{1}
\section{절 제목}\label{sec} ⇒ \sectionline{절 제목}{2}
```

참고

장절 명령 앞과 뒤에 있는 모든 공백과 문자들이 삭제된다는 점을 유념하라.

다음과 같이, 대문자로 시작하는 `\Chapter`나 `\Section` 명령을 사용자가 정의하는 경우도 고려되었다. 그 명령들도 동일하게 처리된다.

```
\let\Chapter\chapter
\let\Section\section
\RenewDocumentCommand\chapter{>{\SplitArgument{1}{|}}m}{\session#1}
\RenewDocumentCommand\section{>{\SplitArgument{1}{|}}m}{\essay#1}
```

`\chapterline`과 `\sectionline`은 변환 명세 파일에 포함된 다음과 같은 지시에 따라 처리된다. 위 예와 같은 경우에 목적에 따라 다른 치환을 지시해야 할 것이다.

```
\sectionline{\{(.)\}|(.+)\ }{\{(.)\ } }
→ <hr /><h2 id="heading_3">\2</h2>\n<p>\1</p>\n
\sectionline{\{(.)\}|(.+)\ }{\{(.)\ } }
→ <h2 id="heading_3">\2</h2>
```

4.3 상호참조

`\label` 명령이 제목 명령과 같은 줄에 있어야 한다.

```
\section{TITLE}\label{sec_label}
```

`\pref` 명령의 사용은 PDF에서 유용하지만 ePub에서 무의미하다. 양 쪽을 충족시키기 위해 이 문서에서 `\titleref` 명령이 다음과 같이 재정의되었다.

```
\DeclareRobustCommand\TitleRef{
  \@ifstar{\@mem@titlerefnoLink}{\@mem@titleref}
}
\RenewDocumentCommand\titleref{m}{
  \pageref{#1} 페이지 \TitleRef{#1}
}
```

`add_heading_id_tex()` 함수와 `get_tex_label()` 함수에 의해 `\titleref`가 다음과 같이

바뀐다.

```
\titleref{chp}
⇒ \headingref{foo.xhtml}{1}{장 제목}\를
\titleref{sec}
⇒ \headingref{foo.xhtml}{2}{절 제목}\가
```

josa 모듈이 자동조사 매크로를 처리한다.

```
\headingref{foo.xhtml}{1}{장 제목}\를
⇒ \headingref{foo.xhtml}{1}{장 제목}을
```

\headingref가 다음의 치환 지시에 따라 HTML 태그로 바뀐다.

```
\\headingref\{(.+?)\}\{(.+?)\}\{(.+?)\}
→ <a href="\1#heading_\2">\3</a>
\headingref{foo.xhtml}{1}{장 제목}을
→ <a href="foo.xhtml#heading_1">장 제목</a>을
```

4.4 각주

create_endnote() 함수가 각주들을 모아 미주 페이지 endnote.xhtml 를 만든다.

\footnotemark와 \footnotetext를 사용하는 경우에 반드시 번호를 기입해야 한다.¹

```
\footnotemark[1] 또는 \footnotemark[a]
\footnotetext[1]{...} 또는 \footnotetext[a]{...}
```

4.5 색인

색인을 위한 변환 명세 파일 indexer 설정 옵션에 지정되어 있으면, create_index() 함수가 색인 항목들을 모아 색인 페이지 index.xhtml 를 만든다.

```
indexer: _html/indexer.tsv
```

indexer 설정 옵션에 복수의 파일들을 지정하는 것은 허용되지 않는다.

HzGuide 클래스에서 정의된 \term은 다음과 같이 기능한다.

```
\term[뒤에]{단어}[아래]
= \index{뒤에@단어}\index{아래!단어}
```

¹\footnotemark 뒤에 \footnotetext를 쓰는 것이 바람직하다.

이와 같이 색인 매크로들이 여럿인 경우가 고려되었다. 색인을 위한 변환 명세 파일은 다음과 같이 작성되어야 한다.

```
\\index\\{(.+?)\\} → \\IndexEntry{\\1}\\index{\\1}
\\term\\{(.+?)\\} → \\IndexEntry{\\1}\\term{\\1}
```

위치 지정자 @는 지원되지만 !는 지원되지 않는다.

4.6 이미지

이미지 파일들에 대해 다음 규칙들이 지켜져야 한다.

- 파일 확장자를 명시하여야 한다.

```
\\includegraphics{foo.pdf}
```

- 위 예에서 파일 경로를 포함하는 것은 허용되지 않는다. 따라서 `\\graphicspath`를 사용하여 이미지 디렉토리의 경로를 지정하여야 한다.

```
\\graphicspath{{images/}{images/fallback/}}
```

- 설정 파일에 이미지 디렉토리의 경로를 지정하여야 한다.

```
original_image_directory:
- images
- images/fallback
```

변환 명세 파일에 다음과 같이 지정하여 PDF 이미지를 SVG나 PNG로 교체할 수 있다.

```
\\includegraphics.*\\{(.+).pdf\\} 
```

참고

이 문서에서는 `\\includegraphics` 대신 HzGuide 클래스에 정의된 `\\image` 명령이 사용되었다.

4.7 HTML 파일로 ePub 만들기

`html_files` 설정 옵션에 HTML 파일들이 지정되어 있으면, `add_heading_id_tex()` 함수가 아이디 `id`가 없는 heading 태그들을 찾아 아이디를 추가한다. heading 태그에 아이디가 없으면 `toc.ncx`가 올바르게 만들어지지 않는다.

HTML 파일들을 ePub으로 변환하려면 HTML 태그들을 XHTML 태그들로 치환하는 다음과 같은 변환 명세 파일이 필요하다.

```
<!DOCTYPE html> → <?xml version="1.0" encoding="utf-8"?>
<html.+?> → <!DOCTYPE ...>
<br> → <br />
<img(.+?)> → <img\1 />
```

참고

HTML 파일에 대한 충분한 테스트가 이루어지지 않았다.

4.8 소스 코드

이 문서에 많은 레이텍 매크로들과 HTML 태그들이 예시되어 있다. 그것들은 다음과 같은 문제를 일으킨다.

```
\section{TITLE} % 이것은 <h2>로 바뀌어야 하지만

\begin{verbatim}
\section{TITLE} % 이것은 무시되어야 한다.
\end{verbatim}
```

이 문제를 피하는 간단한 방법이 백슬래시 뒤에 폭이 0인 공백 문자 ^{U+200B}를 추가하는 것인데, 할 만한 짓이 아니다.

verbatim 환경은 변환 명세 파일에 의해 <pre> 태그로 바뀔 것이다.

```
\begin{verbatim}
<h2>TITLE</h2>
\end{verbatim}
⇒ <pre><h2>TITLE</h2>
<pre>
```

그러나 verbatim 환경과 다르게 pre 블록에 포함된 HTML 태그들이 무시되지 않는다. 이를 위해 또 다른 변환 명세 파일이 필요하다.

```
escape: _html/escape.tsv
```

escape 설정 옵션에 복수의 파일들을 지정하는 것은 허용되지 않는다.

```
< → &#x3C;
> → &#x3E;
\\verb\{(.+?)\} → \\ → &#x5C; → \verb{ → }
```

```
\\begin\\{verbatim\\} → \\ → &#x5c; → \\end\\{verbatim\\}
```

위 예에서 셋째 줄은 `\\verb`에 포함된 백슬래시를 그것의 16진수 코드인 `\`로 바꾸라는 뜻이다. 마찬가지로 넷째 줄은 `\\begin{verbatim}` 이후에 `\\end{verbatim}`을 만날 때까지 그 사이에 있는 백슬래시들을 16진수로 바꾸라는 뜻이다. 셋째 줄의 넷째 칸과 다섯째 칸은 치환된 문자열의 앞과 뒤에 붙는 문자열인 반면, 넷째 줄의 마지막 칸은 정규 표현식이라는 점을 유념하라.

다음과 같은 형식도 허용된다.

```
\\macro\\{(.+?)\\} → \\ → &#x5c; → \\macro{ → } → \\verb{ → }
```

이것에 의해 `\\macro{\\foo}`가 `\\verb{\foo}`로 바뀐다.

다음의 치환 지시는 `verbatim` 환경에 포함된 빈 줄에 공백 문자를 추가한다. 이것은 문단 나누기를 위한 치환에서 소스 코드를 배제하기 위한 것이다. ?? 페이지 ??를 보라.

```
\\begin\\{verbatim\\} → ^$ → &#x200b; → \\end\\{verbatim\\}
```

참고

위에서 설명한 바와 같이 소스 코드를 처리하는 과정에서 `tmp.@@@`라는 이름의 임시 파일이 생성되었다가 삭제된다.

4.9 문단 나누기

하나의 문단을 `<p>`와 `</p>` 태그로 감싸는 것이 바람직하다. 변환 명세 파일에 포함된 다음의 치환 지시들이 연속된 빈 줄들을 하나로 줄인다. 그다음에 빈 줄 앞과 뒤에 `</p>`와 `<p>`를 붙인다.

```
\\n{3,10} → \\n\\n
(?:<=[a-zA-Z가-힣1-9""'.?])\\n\\n(?:<=[a-zA-Z가-힣1-9""'&]) → </p>\\n\\n<p>
(?:<=[a-zA-Z가-힣1-9""'.?])\\n\\n(?:<=) → </p>\\n\\n
(?:<=>)\\n\\n(?:<=[a-zA-Z가-힣1-9""'&]) → \\n\\n<p>
```

내용에 따라 이 치환 지시들을 손봐야 할 것이다.

4.10 주석

장절 명령이 처리될 때 모든 레이텍 주석이 제거된다. TeX 파일로부터 만들어진 XHTML 파일에는 HTML 주석이 있을 수 없다. 그러나 HTML 파일들로부터 변환된 것들과 `additional` 설정 옵션에 의해 복사된 XHTML 파일들에 포함된 주석들은 마크업 변환이 완료된 다음에 제거된다.

```
<!--  
...  
-->
```

— 5 —

HTML 스타일

CSS 파일들이 `additional` 설정 옵션에 기재되어야 한다.

```
additional:
- file: basic.css
  from: _html
  to: OEBPS
- file: adhoc.css
  from: _html
  to: OEBPS
```

그리고 CSS 파일들이 다음과 같이 `\htmlbegin`에 대한 치환문에 포함되어야 한다.

```
\htmlbegin
→ ...<link rel="stylesheet" type="text/css" href="basic.css"/>
  <link rel="stylesheet" type="text/css" href="adhoc.css"/>...
```

동일한 태그 요소 또는 클래스에 대한 스타일들이 있을 때 앞의 것들이 무시되고 맨 나중의 것이 사용된다.

`\textsc`에 대응하는 HTML 태그가 없다. 해법은 `span` 태그에 클래스 속성을 부여하는 것이다.

```
\textsc\{(.+)\} <span class="smallcaps">\1</span>
```

따라서 CSS 파일에 `smallcaps` 클래스가 정의되어 있어야 한다.

```
span.smallcaps { font-variant: small-caps;}
```

CSS 파일에서 `text-indent` 속성이 들여쓰기 크기를 결정한다.

```
p {
  text-indent: 1em;
  margin-top: 0%;
```

```
margin-bottom: 0%;  
}  
.noindent {  
  text-indent: 0em;  
}
```

\noindent 명령이 다음과 같이 바뀌어야 한다.

```
\\noindent\s? → <p class="noindent">
```