멀티마크다운과 라텍

Nova De Hi

2014년 12월 9일

요 약

이 글은 예전에 썼던《스크리브너와 라텍》의 자매편 격이다. 그 글은 스크리브너라는 특정 앱을 기준으로 멀티마크다운을 사용해서 라텍 문서를 만드는 과정을 설명하고 있었기 때문에 스크리브너의 기능을 소개하는 데 중점을 둔 것이었다. 이 글에서는 스크리브너의 의존하지 않고 멀티마크다운 마크업 자체만으로 라텍을 백엔드 프린터로 활용하는 방법에 대하여 기술하려 한다. 한편, 이 문서는 `mmd-oblivoir` 라는 멀티마크다운을 위한 한글 라텍 지원 파일의 사용법 설명을 겸한다.

《스크리브너와 라텍》은 물론이고 내가 번역한《멀티마크다운 문법》이라는 문서를 함께 읽는 것이 좋다. 두 문서에 대한 정보는 참고 문헌을 보면 된다.

차 례

[1] https://creativecommons.org/licenses/by-nc-nd/2.0/kr/

1 서론

"스크리브너와 라텍" [1]이라는 글을 쓴 적이 있다. 이 글은 2014년 초 고려대학교에서 했던 강의를 토대로 작성한 것이었고 라텍 작업을 스크리브너라는 앱이 제공하는 멀티마크다운 신택스를 이용해서 해보자는 제안을 담고 있는 것이었다.

스크리브너는 정말 훌륭한 프로그램이다. 특히 단행본이나 논문과 같은 치밀하고 긴 글, 많은 자료 조사가 필요한 글을 쓰는 데는 이보다 더 나은 툴을 본 적이 없다. 그러나 특히 라텍과 이 앱을 연결하여 생각한다면 본질은 "스크리브너"에 있는 것이 아니라 "멀티마크다운"이라는 마크업 언어에 있는 것이다. 따라서 스크리브너에 의존하지 않는 멀티마크다운 자체와 라텍 작업의 관계를 더 자세히 알아보는 것은 별도의 일일 수도 있다.

이 글은 라텍의 프론트엔드, 또는 "전작업 도구"로 멀티마크다운에 대해 알아본다. 주요한 관심사는 멀티마크다운이 문서 작성에 얼마나 유용한 도구인가하는 것과, 라텍을 출력 포맷으로 하기로 했을 때 그 뒷감당(!)을 어떻게 할 것인가에 대해, 특히 한글 문서를 중심으로 몇 가지 필자의 경험에 기초한 이야기를 하고자 한다.

2 전이해

2.1 멀티마크다운이 해줄 수 있는 것

멀티마크다운은 John Gruber의 마크다운 언어의 확장판으로 Fletcher Penney가 만든 것이다. 여러 종류의 경량 마크업 언어(Lightweight Markup Language[2])가 수없이 많지만 그 중에서 문제 삼고자 하는 것은 Markdown[3]과 MultiMarkdown[4]이다.

Fletcher Penney는 멀티마크다운에 대해서 "80%의 사람들이 작성하는 80% 정도의 문서에 적합한" 도구라고 소개하고 있다. 이 말은 멀티마크다운이 LaTeX이나 Microsoft Word와 같이 최종 출력물을 즉시 얻을 수 있는 최종판 제작 도구가 아니라, 그런 최종판을 편집하기 전에 문서의 내용을 구성하는 도구임을 명심해야 한다는 의미가 된다. 다르게 말하면, 멀티마크다운은 HTML, LaTeX, ODF 등 어떤 출력물을 생산해내는 도구인데, 최종판은 그런 도구를 직접 편집할 수 있는 앱으로 하라는 것이다. 즉 대단히 구체적인 최종 출력물의 외관은 멀티마크다운이 직접 책임지지 않는다. 예를 들어보자.

1. 멀티마크다운 언어를 사용하여 문서의 모든 내용을 구성한다.
2. 여기에는 예를 들면 그림이나 표, 문헌 목록 등이 다 들어갈 수 있다.
3. 그러나 이것으로 pdf 출력을 해보면 최종판이라고 하기에는 2% 부족하다.
4. LaTeX으로 export한 결과에 대하여 (이번에는 내용은 건드릴 필요가 없으므로) 폰트나 정렬 방식, 그림의 배치 등을 구체적으로 손을 본다. 이게 나머지 2%.
5. 원하는 최종 출력을 pdf로 얻을 수 있다.

이런 식의 작업을 하게 해주는 도구라는 것이다.

멀티마크다운으로 모든 것을 하려 해서는 안 된다. 그런 "모든 것"에는 고려할 점이 너무 많이 들어간다. 그림을 예로 들자면 멀티마크다운이 관심을 가지는 것은 "여기에 이러한 그림이 있다"는 것까지이다. 그 그림이 문단 내부에 배치될 것인지 수평으로 배열될 것인지 등등에는 관심 없다. 그런 것까지 멀티마크다운 마크업으로 다 하려면 차라리 라텍 코딩을 그냥 하는 편이 훨씬 낫다. 멀티마크다운의 철학은 이러한 외관에 관련된 속성은 최종 출력물을 조성할 때 가서 생각해도 된다는 것이다. 글을 쓸 때는 말 그대로 "여기 이 그림"으로 충분한 것이다.

[2] http://en.wikipedia.org/wiki/Lightweight_markup_language
[3] http://en.wikipedia.org/wiki/Markdown
[4] http://en.wikipedia.org/wiki/MultiMarkdown

2.2 사례

한 예를 들어보겠다. 나는 얼마 전에 Nicola Talbot의 "라텍 최소 예제 작성법"[5]이라는 문서를 번역했다. 이 작업을 어떻게 진행했는가?

1. mmd 에디터[6]로 원문을 보면서 번역 작업을 진행한다.
2. Marked 2[7]라는 앱을 이용하여 이 mmd의 HTML 결과물을 보면서 교정까지 마친다.
3. 이것을 `.tex` 파일로 변환(export)한다.
4. 일단 컴파일해본 다음 수정할 부분을 찾는다.
    a) `\href{A}{B}\footnote{…}` 부분을 정규식으로 모두 변환하여 `\footnote` 부분을 없앤다. 사실 이것은 취향의 문제인데 이 문서에서는 본문에 링크가 걸리는 것으로 충분하다고 판단하였다. 각주가 너무 많아지는 것이 번거롭기도 했고. 아무튼 이것을 바꾸는 것이 어렵지 않다는 것이다.
    b) 나중에 `\LaTeX`으로 바꾸기 위하여 원래 `@LaTeX@`으로 표현했던 부분을 한꺼번에 바꾼다.[8]
    c) `verbatim` 환경 다음에 빈 줄이 없어야 하는 부분 두어 곳을 찾아 수정한다.
    d) 사람 이름에 쓰인 period에 추가 공간이 생기지 않도록 수정한다.
5. 이제 최종 출력물이 완성되었다. pdf 파일을 업로드한다.

보는 대로, 그다지 번거롭지 않다. 특히 `.tex` 파일을 약간(!) 수정하는 것이 귀찮아 보일지도 모르지만 정규식 찾기/바꾸기를 이용하면 정말 간단히 한 줄의 명령으로 모두 해결된다.

MMD는 라텍-문맹을 위한 툴이 아니다. 오히려, 라텍을 잘 아는 사람에게 정말로 유용한 도구이다.

3 설치

설치(installation)는 모든 것의 시작이다. MMD로 (한글) 라텍 문서를 작성하려면 다음과 같은 것이 설치되어야 한다.

1. TEX 시스템. 즉 TEX Live
2. `multimarkdown` 실행 파일과 유틸리티

---

[5] http://wiki.ktug.org/wiki/wiki.php/KTUGExtDocArchive

[6] 사실 MMD 에디터는 그냥 플레인 텍스트(= 프로그래밍) 에디터이다. 나는 맥에서 ByWord라는 프로그램을 이용했다. 아무 생각없이 MMD로 글쓰기에 딱 좋은 정도의 별 복잡한 기능 없는 마크다운 에디터이다.

[7] 일종의 마크다운 미리보기 앱이다. 써보면 매우 편리하다. 다만 유료인 점이……. http://marked2app.com/

[8] mmd에서 `\LaTeX`이라고 써넣기는 어렵다. 웬만하면 그냥 LaTeX으로 좋지만 굳이 로고를 그대로 찍겠다면 먼저 mmd 코딩 시에는 @LaTeX@이라고 해두었다가 나중에 `.tex` 파일에서 일괄변환하는 것이 가장 좋다.

a) `multimarkdown`

b) MMD-Support

3. MultiMarkdown LaTeX Support 파일
4. 한글 mmd-LaTeX 지원 파일

이 가운데 TeX Live 설치는 플랫폼 상관 없이 간단히 진행할 수 있으므로 이 글에서 더 자세히 다루지 않겠다. KTUG의 설치 안내 페이지[9]를 보면 충분하다.

3.1 TnXTeX

Windows용 TnXTeX[10]은 multimarkdown을 즉시 사용할 수 있는 형태로 제공한다. 이것 저것 귀찮다면 이것을 이용해보자.

3.2 MultiMarkdown

멀티마크다운은 `multimarkdown(.exe)`이라는 실행 파일 하나로 이루어져 있다. 원칙적으로 이 파일 하나만 있으면 되는 것이지만 우리는 `mmd2tex`이라는 배치파일(셸 스크립트)을 주로 쓸 생각이므로 이에 관련된 파일들도 함께 설치하는 것이 좋다. 즉 MultiMarkdown (MMD)-Support라는 것을 추가로 설치해주려고 한다.

설치에 필요한 파일은 모두 Fletcher Penney의 MultiMarkdown 사이트[11]에서 구할 수 있다.

Mac OS X

맥에서 설치가 아마 가장 쉬울 것이다. 다음 두 파일을 다운로드받아서 보통 하듯이 그냥 설치하면 끝.

- Mac Installer[12]
- Mac Support[13]

Linux

리눅스에서는 직접 빌드하는 것이 가장 빠르고 확실하다.

적당한 디렉터리에서 다음 명령을 차례로 실행하자.

---

[9] http://wiki.ktug.org/wiki/wiki.php/TeX%20Live%EC%99%80%20ko.TeX%EC%9D%98%20%EC%84%A4%EC%B9%98

[10] http://wiki.ktug.org/wiki/wiki.php/TnXTeX

[11] http://fletcherpenney.net/multimarkdown/

[12] http://files.fletcherpenney.net.s3.amazonaws.com/MultiMarkdown-Mac-4.6.zip

[13] http://files.fletcherpenney.net.s3.amazonaws.com/MultiMarkdown-Support-Mac-4.6.zip

```
$ git clone git://github.com/fletcher/MultiMarkdown-4.git
$ cd MultiMarkdown-4
$ git clone git://github.com/fasterthanlime/greg.git
$ cd ..
$ make
```

현재 폴더에 `multimarkdown`이라는 실행 파일이 생기면 정상이다. 이 파일을 `/usr/local/bin/`으로 옮긴다.

```
$ sudo cp ./multimarkdown /usr/local/bin/
```

그리고 `scripts/` 아래 있는 스크립트들도 (필요하다면 원하는 것만) 옮긴다.

```
$ sudo cp scripts/mmd2tex /usr/local/bin/
```

이제 MMD-Support를 설치하자.

```
$ git clone git://github.com/fletcher/MMD-Support.git
```

`MMD-Support` 안에 들어가면 `bin` 폴더와 `XSLT`가 있다. 이것을 `/usr/local/XSLT`와 `/usr/local/bin/`으로 복사하면 된다. 여기서는 `mmd2tex-xslt` 스크립트만 옮기는 예를 보이겠다.

```
$ sudo mkdir -p /usr/local/XSLT
$ cp XSLT/* /usr/local/XSLT/
$ cp bin/mmd2tex-xslt /usr/local/bin/
```

Windows

Windows는 Installer가 제공된다.

- Windows Installer[14]

Portable 버전도 있으므로 원한다면 해당 파일을 이용해도 좋다.

Windows에서 MMD-Support를 설치하는 것은 꽤 복잡한 문제이다. 셸 스크립트와 perl 스크립트가 실행되게 해야 하는 문제도 있고… 따라서 MMD-Support는 잊어버리기로 한다. 윈도우즈가 그렇지 뭐.

[14] http://files.fletcherpenney.net.s3.amazonaws.com/MultiMarkdown-Windows-4.6.zip

3.3 LaTeX 지원 파일

peg-multimarkdown-latex-support[15]라는 이름이 붙은 파일군을 자신의 TeX 시스템에 설치해두는 것이 좋다. 몇 가지 유용한 파일이 포함되어 있다.

명령행 `git` 툴을 사용할 수 있다면

```
git clone git://github.com/fletcher/peg-multimarkdown-latex-support.git
```

`peg-multimarkdown-latex-support` 폴더를 `$TEXMFHOME/tex/latex/` 아래 가져다 두는 것으로 충분하다.

혹시 `git` 툴을 사용할 수 없다면 위에 보인 URL에 접속하여 `Clone in Desktop`이나 `Download ZIP`을 이용하여 내려받을 수 있다.

아무튼, `$TEXMFHOME`의 위치는 `kpsewhich --var-value=TEXMFHOME` 명령이 보여주는 경로인데 Windows라면 자신의 홈 폴더 아래 `texmf`, 리눅스에서는 `~/texmf`, 맥에서는 `~/Library/texmf`이다. 이 폴더가 없으면 만들면 되고 이 아래에 TDS에 맞추어서 두는 파일에 대해서는 `mktexlsr`하지 않아도 된다.

3.4 mmd-korean

MMD-Korean이란 필자가 작성한 한글 LaTeX 지원 mmd 파일의 묶음이다. 여기에는 세 종류의 mmd 지원 파일이 포함되어 있다.

1. mmd-oblivoir. `oblivoir` 클래스 문서를 작성하게 한다.
2. mmd-korean. chapter가 있는 큰 규모의 문서를 작성한다.
3. mynovel. 소설 쓰기를 위해 만든 클래스 `mynovel.cls`를 이용하는 mmd 파일들이다.

한글 라텍에 대하여 잘 알고 있어서 스스로 헤더 파일을 만들 수 있다면 이런 보조 파일은 필요없다. 그러나 이 글의 예시문을 위해서는 위의 파일을 설치해야 하므로, 책읽기의 낙원[16]에서 제공하는 mmd-korean.zip[17]을 다운로드하자. 압축을 풀어서 `$TEXMFHOME/tex/latex/` 아래 가져다 두면 된다.

mynovel을 사용하려면 약간 설정이 필요한데, TeX Live의 `texmf.cnf`[18]에 다음 한 줄을 추가해 주어야 한다. 이 글에서는 mynovel에 대해 언급하지 않을 것이므로 무시해도 좋다.

---

[15] https://github.com/fletcher/peg-multimarkdown-latex-support

[16] http://doeun.blogspot.kr/2014/10/mmd-korean-multimarkdown-latex.html

[17] https://www.dropbox.com/s/wosuxng9h14sbce/mmd-korean.zip?dl=1

[18] 윈도우즈에서는 `c:\usr\texlive\2014\texmf.cnf`이고 맥이나 리눅스에서는 `/usr/local/texlive/2014/texmf.cnf`일 수 있다. 이 파일의 위치는 `kpsewhich texmf.cnf` 명령으로 찾는다. 단 `texmf-dist/web2c/texmf.cnf`와 같은 파일을 수정하면 안 된다.

```
max_in_open = 20
```

4 첫 문서와 실행과정 이해

준비가 끝났으므로 이제 첫 번째 문서를 하나 작성해보고 이것이 어떤 과정을 거쳐서 LaTeX 파일이 되어 pdf로 출력되는지 그 전체 과정을 이해해보기로 하자.

4.1 첫 번째 샘플 문서

적당한 에디터를 열어서 다음과 같은 내용을 적어넣고 저장한다. 저장 시 인코딩은 **반드시** UTF-8으로 한다.

```
latex input: mmd-oblivoir-header
base header level: 3
title: My Title
author: author
latex input: mmd-oblivoir-begin-doc

# 첫 문서
이제 멀티마크다운 첫 문서를 작성하였습니다.
```

이 파일을 `myfirsttest.mmd`라는 이름으로 저장하자. 멀티마크다운 파일의 확장자는 `md`, `mmd`, `text`, `mdown` 등 어떤 것으로 해도 좋지만 나는 `mmd`로 하는 것을 선호한다.

파일이 저장된 폴더에서 명령행을 열고 다음 명령을 실행하자.

```
$ mmd2tex myfirsttest.mmd
```

아무 반응 없이 종료될 것이다. 그리고 같은 폴더에 `myfirsttest.tex`이 생긴다. 이 파일을 TeXworks나 TeXshop으로 열어서 검토하면 그림 1과 같은 모습으로 되어 있다.

조판 명령을 실행하여 보라. 에러 없이 컴파일되고 pdf가 생성되었는가? 참고로, `pdflatex`으로 컴파일하면 알 수 없는 에러가 발생할 것이다. 이 에러가 발생하지 않도록 하는 방법에 대해서는 나중에(pdflatex에서 에러가 발생하지 않게 하는 법 (subsection 9.1)을 볼 것) 논의하겠다. 여기서는 `xelatex`으로 컴파일하면 아무 문제가 없다는 것을 확인하고 지나가자. **한글 문서는 XeLaTeX이 표준이다.**

4.2 샘플 문서의 이해

첫 번째 샘플 문서 `firsttest.mmd`는 크게 보면 두 부분으로 이루어져 있다. 맨 첫 줄에서 다섯 번째 줄까지 `key: value` 형식으로 적어넣은 부분을 "메타데이터(meta-data)"라고 한다. 메타데이터는 멀티마크다운의 독특한 형식이고 HTML, LaTeX, OpenDocument 등

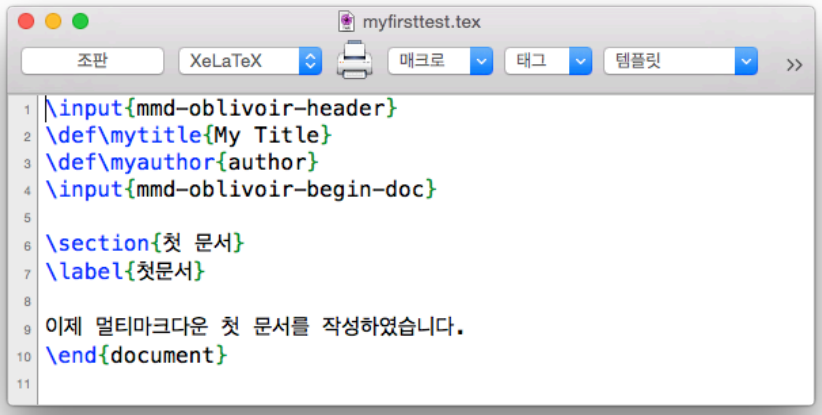


그림 1: 샘플 문서의 변환 결과

여러 포맷으로 출력하는 것에 대응하여 문서를 제어하는 기본 정보의 역할을 한다. 우리 관심사는 LaTeX으로 변환하는 것뿐이므로, 다른 포맷에 관한 사항은 언급하지 않겠다.

메타데이터 부분은 반드시 MMD 문서의 맨 처음에 와야 한다. 그 앞에 빈 줄이 있어도 안 된다. 그리고 모든 메타데이터 키는 각 행의 첫 번째 열, 즉 행머리에서부터 시작해야 하고 그렇지 않으면 멀티마크다운이 메타데이터 키로 인식하지 않는다. 키의 입력이 끝나면 :과 스페이스 하나 (이상)을 적고 그 값(value)을 적어넣으면 된다. 만약 value가 두 줄 이상이라면 두 번째 줄부터는 들여쓰기를 하여 meta-data key가 아니라는 것을 알려주어야 한다.

제1행과 제5행에 `latex input:`이라는 메타데이터 키가 지정되었다. `latex input` 메타데이터 부분은 멀티마크다운에 의하여 `\input{…}` 문으로 바뀐다. 즉 `latex input: mmd-oblivoir-header`라는 것은 `\input{mmd-oblivoir-header}`로 변환되어 실제 LaTeX 컴파일시에 `mmd-oblivoir-header.tex`이라는 파일을 찾는다.

title과 author는 이름 그대로인데 `mmd-oblivoir-begin-doc`이라는 파일에 이 값들을 어떻게 처리할 것인지 정의되어 있다. 즉, `\maketitle`의 정보로 사용하게 된다.

`base header level`에 대해서는 약간 주의가 필요하다. 아주 간단히 말하자면, MMD의 제1수준 헤더–즉 HTML에서 <h1>으로 바뀌는 것으로 # 하나로 표현되는 표제–가 어떤 수준의 장절 표제 명령에 대응하도록 하는가를 결정하는 값이다. 이 값이 1이면 #은 `\part`, ##은 `\chapter`와 같이 된다. 이 예제에서 3으로 한 것은 결국 #(<h1>)을 `\section`으로 만들라는 지시인 것이다. 따라서 그 아랫수준의 장절 표제들은 ##이 `\subsection`, ###이 `\subsubsection` 순으로 옮겨지게 한다. 당연한 것이지만

`mmd-korean`은 `\chapter`가 있는 문서를 작성하는 데 사용하는 것이므로 이 값을 2로 지정해야 할 것이다.

메타데이터를 지시한 후 한 줄을 띄우고 그 뒤에서부터 문서를 작성하면 된다. 여기서는 장절 표제 하나를 두고 한 문장을 써보았다.

5 텍스트 마크업 규칙

아마도 세상에서 제일 쉬운 마크업 언어 중의 하나일 멀티마크다운 마크업에 대해서 알아보자. 멀티마크다운 자체는 이른바 "경량 마크업 언어([Lightwight Markup Language])" 중의 하나로서 John Gruber가 만든 Markdown[19]을 확장한 것이다.

Fletcher Penney 사이트의 다음 몇 곳은 한 번쯤 보아야 한다.

- MultiMarkdown이란 무엇인가? http://fletcherpenney.net/multimarkdown/ 이 페이지는 dokenzy가 번역한 것 http://blog.dokenzy.com/archives/821과 필자가 옮긴 것 http://doeun.blogspot.kr/2014/02/fletcher-penney-multimarkdown.html이 있다.
- MultiMarkdown 사용자 설명서 http://fletcher.github.io/MultiMarkdown-4/ 이 내용의 일부를 필자가 번역한 것은 http://doeun.blogspot.kr/2014/12/blog-post.html. [2]
- MultiMarkdown Syntax Cheat Sheet https://rawgit.com/fletcher/human-markdown-reference/master/index.html

기본적인 것은 위의 몇 곳에 다 있으므로, 여기서는 특히 한글 문서 작성과 관련하여 주의할 것을 중점으로 적어보겠다.

5.1 장절 표제

멀티마크다운은 다음과 같은 형식의 장절 표제를 지원한다.

```
# 제 1 수준 헤더 #

## 제 2 수준 헤더 ##
```

마크다운에서 온 다음의 형식도 일부 지원되는 것으로 보이나, 되도록 위의 방식을 쓰는 것이 좋다.

```
제 1 수준 헤더
==============

제 2 수준 헤더
--------------
```

[19] http://en.wikipedia.org/wiki/Markdown

헤더를 나타내는 # 문자는 해당 행의 첫 번째 글자여야 한다. 마지막에 오는 #은 없어도 상관없다. 즉, 다음처럼 써도 된다.

```
# 제 1 수준 헤더
```

제 1 수준 헤더가 \part인지 \chapter인지 \section인지는 앞서 설명한 대로 메타데이터의 base header level 값을 따른다. 이 메타데이터가 없으면 \part부터 시작한다.

상호참조를 위한 label 즉 anchor는 다음처럼 임의로 부여할 수 있다.

```
# 제 1 수준 헤더 [header] #
```

이 label이 주어지지 않으면 헤더의 텍스트를 normalize하여 label로 사용한다. 스페이스를 전부 없애고 문자만으로 된 anchor를 만든다. 즉 별도의 anchor를 부여하지 않았을 때 label은 \label{제1수준헤더}가 된다.

5.2 텍스트 강조

표준적인 마크다운과 멀티마크다운은 모두 * 문자를 이용한 강조와 _ 문자를 이용한 강조를 지원한다. 강조를 나타내는 * 문자는 반드시 그 앞에 다른 문자와 스페이스 없이 붙으면 안 된다. 한 개는 이탤릭체이고 두 개는 볼드이다. 다시 말하면, *강조유형 1*은 \emph{강조유형1}로 변환되고 **강조유형2**는 \textbf{강조유형2}로 바뀐다. 어떤 변형 마크다운은 세 개짜리도 지원하는 게 있지만 멀티마크다운은 두 개까지만 지원한다.

라텍을 백엔드로 생각하고 작업한다면 혼선을 피하기 위해서 _ 문자보다 *를 사용하는 것이 낫다. 왜냐하면 _ 문자는 LaTeX에서 하첨자를 나타내는 부호로 쓰이고 수학 모드가 아니면 에러를 보이기 때문이다. 변환이 성공적으로 이루어지지 않을 경우를 대비하여 _ 방식보다 * 방식을 선호하는 것이다.

```
*italicized*, _italicized_, **bold**, __bold__
```

italicized, *italicized*, **bold**, **bold**

한글 문서에서 \emph가 어떻게 구현되게 하느냐 하는 것은 클래스의 설정에 달려 있다. mmd-oblivoir의 디폴트는 영문자에는 이탤릭이 적용되지만 한글은 변화없이 나타날 것이다.[20] 이 설정을 수정하는 문제는 한글 이탤릭체 (subsection 9.2) 절에서 설명한다.

LaTeX의 \texttt로 옮겨지는 인라인 코드는 꽤 쓸모가 많다. 이것은 backquote문자(`)로 둘러싸서 표현한다.

```
`some code`
```

[20] oblivoir 클래스의 기본값.

5.3 문장 부호

텍스트 입력 시의 부호와 관련하여 몇 가지 알아보자.

유보문자

LaTeX의 입장에서 특히 좋은 점은 유보 문자들에 대한 걱정 없이 그냥 문자만 쳐넣어도 된다는 것이다. 예를 들어

```
& # _ ^ % $ \ { } ~
```

이것이 아무 에러를 일으키지 않고 적절하게 변환되어 표시된다: & # _ ^ % $ \ { } ~.

한편, 멀티마크다운이 사용하는 몇 개의 마크업 문자들을 바로 표시하려면 \를 앞에 붙여주면 된다. *, \~, \[, \] 등. 이와 관련하여 한 가지를 지적하고자 한다. 예를 들어 * 문자가 나온 경우에 그것이 멀티마크다운이 판단하기에 마크업으로서 적절한 위치, 즉 (1) 불릿붙은 리스트를 나타내는 행머리의 *인가, (2) emph-강조로서 스페이스 뒤에 온 후에 스페이스 없이 몇 글자가 오고 다시 *가 출현하는가를 판단하여 각각 리스트로 만들거나 emph-강조로 처리할 것이다. 그렇지 않고 해석할 수 없는 위치에 있다면 그대로 별표(*)를 찍어준다.

하이픈과 대시

이것은 LaTeX의 경우와 같다.

- -: 하이픈
- --: en-dash
- ---: em-dash

범위를 나타내는 데 LaTeX에서는 en-dash를 쓰지만 우리 글로 된 문헌에서는 물결표(~)를 쓰는 때도 많다.

상첨자와 하첨자, 틸데문자

LaTeX에서 상첨자와 하첨자는 수학 모드에서만 가능한 것이 원칙이다. 텍스트 모드에서 첨자를 구현하려면 \textsuperscript와 같은 마크업이 필요하다. MMD는 ^와 ~로 상첨자와 하첨자를 텍스트 모드에서 쓸 수 있다. 가^나, 다~라. 가^{나}, 다$_{라}$. 단 두 단어 이상의 첨자는 바로 지원하지 않는다. 긴 첨자를 다음과 같이 쓸 수는 있지만 스페이스는 없어야 한다.

```
단어^긴첨자^
단어~긴첨자~
```

단어^긴첨자 단어_긴첨자

이 기능은 예컨대 `m^2` (m^2)과 같은 단위 표시에서 편리할 것이다.

한글 문서에서 이것은 한 가지 재미난 결과를 가져온다. `\~`(틸데) 문자는 멀티마크다운이 하첨자를 식자하게 하는 마크업 부호이다. 예를 들면 `a~2` a$_2$. 이와 같이 마크업 부호로 인식되는 것은 이 문자 전후에 스페이스가 전혀 없을 때인데, 그러면 뒷글자를 앞글자의 하첨자로 식자하게 한다. 즉 `\textsubscript`로 변환된다. 그런데 우리는 이것을 이따금 범위를 나타내는 물결표 대신 쓰는 경우가 있다. 2~3. 이런 의도로 쓰려 한다면 멀티마크다운이 헷갈리지 않게 `\~`로 써주는 것이 좋다.

```
234\~56
```

234~56. LaTeX에서는 `$\sim$`으로 식자된다.

따옴표

LaTeX 마크업의 가장 큰 특징 중의 하나가 여는 따옴표와 닫는 따옴표가 다르다는 것이다. 그런데 MMD에서는 ‘ 문자가 짧은 코드를 쓰는 데 사용되고 있기 때문에 따옴표 입력에서 주의하여야 한다.

작은 따옴표는 `'text'`와 같이 마크업하고 큰 따옴표는 `"text"`와 같이 하면 되어야 하는 것이 원칙이다. 멀티마크다운이 이것을 “smart”하게 변환해주기로 되어 있다. 그런데, 한글 문서는 알파벳 문자로 이루어지는 문서와 다른 점이 있다. 닫는 따옴표 뒤에 당연히 스페이스가 오는 서양 문서와 달리 우리 글로 된 문서는 닫는 따옴표 뒤에 **조사**가 이어붙는 경우가 많다는 점이다. 그래서 한글 멀티마크다운 문서에서 따옴표의 여닫기를 mmd가 제대로 파싱하지 못하는 경우가 생긴다. 구체적인 예를 들면 다음과 같다.

1. 여는 작은 따옴표가 구현되지 않는 경우
2. 한글과 영문을 섞어쓰는 문서에서 영문자 부분으로 넘어가는 여는 큰 따옴표가 구현되지 않는 경우
3. 대화 내에 행바꿈(개행)이 있으면 따옴표 여닫기가 실패하는 경우

이 문제를 해결하는 가장 좋은 방법은, 에디터가 따옴표를 *smart*하게 처리해주는 것이다. 이런 기능을 가진 에디터가 꽤 많다. 스크리브너도 그 중의 하나이다. TeXworks의 “똑똑한 따옴표” 기능을 이용해도 된다.

또는, 어차피 LaTeX 문서가 될 테니 (여는 따옴표를) LaTeX 식으로 입력하는 방법도 있다. 예를 들면 다음과 같다.

```
\`\`텍스트''
```

이것은 LaTeX에서 원하는 결과를 얻게 할 것이다. 특히 웬만한 똑똑한 에디터도 따옴표 여닫기로 인식하지 못하는 세 번째 경우에는 이 방법이 유일한 해결책이 된다.

줄임표

온점 세 개는 MMD가 `\ldots`로 바꾸어 준다. 줄임표(ellipsis)에 해당한다. xetex-ko 패키지는 한글 상황에서 `\ldots`가 오면 우리 말의 "줄임표"의 반(…)에 해당하는 부호로 식자한다. 이 부호의 모양은 식자하고 있는 폰트에 달렸다. 우리 말 어문규정의 줄임표는 점이 여섯 개여야 하므로 이것을 두 번 쓴다…….

5.4 코드 블럭

멀티마크다운의 "코드 블럭"이란 LaTeX의 `verbatim` 환경으로 변환되는 것을 말한다. 짧은 행중의 코드는 `로 입력하면 되고 한 줄 이상의 긴 코드는 코드 블럭으로 표현한다.

코드 블럭을 표현하는 방법은 두 가지가 있는데, 둘 다 LaTeX으로 변환했을 때의 결과는 같다.

1. 행 머리에 3 이상의 빈 칸을 두어 들여쓰기 한다.
2. backquote 문자(`)를 3 내지 5개 두어서 코드 블럭의 범위를 표시한다.

첫 번째 것을 "들여쓰기 코드 블럭"이라 하고, 두 번째 것은 "fenced 코드 블럭"이라 하는데, 편리한 것을 쓰면 된다. fenced 코드 블럭은 멀티마크다운에 있는 것이다.

```
    들여쓰기 코드 블럭은
    이런 식으로 표현합니다.
```

````
```
fenced 코드 블럭은
이런 식으로 표현합니다.
```
````

fenced 코드 블럭을 코딩할 때는 세 개의 back quote 글자 앞에 한 줄을 비우는 것이 좋다.

listings와 코드 블럭

fenced 코드 블럭에 언어 옵션을 주어서 다음과 같이 코딩하면,

````
``` tex
\begin{document}

\section{Section}
```
````

이 부분은 verbatim이 아니라 다음과 같이 변경된다.

```
\begin{lstlistings}[language=tex]
\begin{document}

\section{Section}
\end{lstlistings}
```

즉, listings 패키지를 사용하도록 바꾸는 것이다. `mmd-oblivoir`에 이에 대한 정의가 없으므로 다음과 같은 내용으로 `mypreamble.tex`을 작성하여 `latex input`하도록 하자.

```
\usepackage{listingsutf8}
\lstset{escapeinside=~~}
```

두 번째 줄은 한글을 표현하기 위해서 필요하다. 즉 한글이 이 영역 안에 들어간다면 ~ 문자로 시작과 끝을 표시하도록 한 것이다. 원한다면 다른 문자로 바꾸어도 된다. 예컨대 ~~ 이라면 소스 안에서 이 문자를 다른 용도(원래의 의미인 unbreakable space)로 사용해서는 안 된다.

```
``` tex
\usepackage{listingsutf8}
\lstset{%
    escapeinside=~~,
    basicstyle=\ttfamily\small,
    keywordstyle=\color{blue}\bfseries,
    numbers=left,
    numberstyle=\tiny,
    tabsize=4,
}
```
```

```
\usepackage{listingsutf8}
\lstset{%
    escapeinside=,
    basicstyle=\ttfamily\small,
    keywordstyle=\color{blue}\bfseries,
    numbers=left,
    numberstyle=\tiny,
    tabsize=4,
}
```

listings 패키지를 이용하여 예약어에 색깔을 입히는 등 여러 가지를 할 수 있다. 이 문제와 관련하여 자세한 사항은 listings 패키지 문서를 참고하라. 모든 세팅은 `mypreamble.tex`에서 하여야 한다는 사실만 주의하면 원하는 대로 verbatim 텍스트를 장식할 수 있다.

코드 블럭의 응용

코드 블럭은 그 블럭 안의 텍스트가 있는 그대로 전달된다는 특징이 있다. 이를 이용하여 몇 가지 응용을 해볼 수 있다.

- 어차피 이 범위의 텍스트는 verbatim으로 전달된다는 것을 이용하여 verbatim 환경을 재정의하거나
- XSLT 파일을 변형하여 verbatim이 아니라 이를테면 verse로 전달되게 하는 것 등이 시도되었다.

책읽기의 낙원 블로그에 소개된 다음 글들은 그러한 시도를 보여주고 있다.

1. 코드 블럭을 운문(verse) 환경으로[21]
2. 코드 블럭을 minted로[22]

일단 멀티마크다운에서 verbatim으로 넘겨주면 그걸 가지고 어떻게 디스플레이할 것인지는 LaTeX 쪽에서 해결할 문제이다. 스타일이나 클래스를 수정함으로써 원하는 어떤 형식으로든 출력이 가능하다. 그리고 이것은 MMD가 아니라 LaTeX 스타일/클래스를 수정함으로써 이루어지는 것이다.

mmd-oblivoir를 그대로 쓰는 경우, 다음과 같은 코드를 추가하여 모든 verbatim에 박스를 두르는 방법도 있다.

```
\let\ORGverbatim=\verbatim
\let\endORGverbatim=\endverbatim
\let\verbatim=\boxedverbatim
\let\endverbatim=\endboxedverbatim
```

위의 방법은 실제로 내가 라텍 최소 예제 작성법[23]을 작업하면서 썼던 것이다. 즉, 위의 네 줄을 myverbatim.tex으로 ($TEXMFHOME/tex/latex/ 아래에) 저장하고 메타데이터를 다음 순서로 하면 간단히 구현된다.

```
latex input: mmd-oblivoir-header
latex input: myverbatim
latex input: mmd-oblivoir-begin-doc
```

5.5 블럭 인용

멀티마크다운은 "블럭 인용"(>로 표시된 문단)을 quote 환경으로 전달해준다.

> 인용문

[21] http://doeun.blogspot.kr/2014/02/mmd-verse.html
[22] http://doeun.blogspot.kr/2014/11/mmd-minted.html
[23] http://wiki.ktug.org/wiki/wiki.php/KTUGExtDocArchive

이것은 다음과 같이 넘어온다.

```
\begin{quote}
인용문
\end{quote}
```

블럭 인용은 중첩될 수 있는데, 어차피 LaTeX의 quote도 중첩 가능하므로 결과는 같다. 인용문단이 어떤 모양으로 최종 pdf에 나타나느냐는 클래스/스타일에서 정의하기 나름이다.

> MultiMarkdown으로 작성하면 문서의 내용과 구조를 모양으로부터 분리할 수 있다. 장절제목이나 문단 간격같은 스타일을 만드는 데 신경쓰지 않고 글쓰기에만 집중할 수 있다. 그리고 좀 더 생각해보면, 하나의 플레인 텍스트 문서는 여러 출력 포맷으로 변환할 수 있다. 일일이 문서 전체나 포맷을 다시 쓸 필요가 없다. 더 좋은 점은, 좋은 HTML문서나 LaTeX 명령을 만들기 위해 "컴퓨터 언어"로 쓸 필요가 없다. 여러분은 글만 쓰면 된다. 나머지는 MultiMarkdown이 해준다.[24]

위의 문단이 인용문단이다.

멀티마크다운이 넘겨주는 인용문단이 quotation이 아니라 quote이기 때문에 예를 들어 둘 이상의 문단은 다음과 같은 방식이 된다.

```
\begin{quote}
첫째 문단
\end{quote}
\begin{quote}
둘째 문단
\end{quote}
```

이 결과가 마음에 들지 않는다면, XSLT를 수정하여 quote가 아니라 quotation으로 넘어오게 하는 방법이 있지만 이것은 좀 무리해 보이고, 권장하고 싶은 것은 마지막에 가서 .tex 파일을 최종적으로 수정하라는 것이다.

5.6 리스트 문단

LaTeX에서 "리스트 문단"이라 함은 itemize, enumerate, description이 대표적인 것이다. 실제로는 quote, quotation, verse 등도 모두 리스트 문단이지만 이들은 글머리표가 붙지 않고 문단 모양만 달라지는 것이므로 내부적으로야 어떻게 설계되었든 다른 형식으로 보아도 되겠다.

[24] http://blog.dokenzy.com/archives/821

마크다운 자체가 두 가지 리스트 문단을 제공한다. 즉 "번호붙인 리스트"와 "불릿붙은 리스트"가 그것인데 각각 다음과 같이 마크업한다.

```
1. 우유
1. 빵
1. 치즈
    1. 체다
    1. 까망베르
1. 밥
```

1. 우유
2. 빵
3. 치즈
    a) 체다
    b) 까망베르
4. 밥

```
* 우유
* 빵
* 치즈
    * 체다
    * 까망베르
* 밥
```

- 우유
- 빵
- 치즈
    - 체다
    - 까망베르
- 밥

번호붙은 리스트는 (빈 행 하나가 온 후에) 제일 첫 번째 글자가 숫자(어떤 숫자여도 상관없다)로 시작하면 만들어진다. 이 리스트는 중첩 가능한데 두 번째 수준 리스트는 **탭** 들여쓰기로 구분한다.

이 환경의 끝은 "더이상 리스트가 아님"을 나타내는 데까지 이어진다. 예를 들면,

```
1. 우유

* 빵
* 치즈
```

이렇게 한다고 해서 앞 부분이 `enumerate`로, 뒷부분이 `itemize`로 되는 것이 아니다. 왜냐하면 `*` 빵이라고 한 부분도 앞 문단과 마찬가지로 리스트 문단이므로 멀티마크다운이 리스트 환경을 그 앞에서 끝낼 이유를 찾지 못했기 때문이다. `enumerate`이냐 `itemize`냐는 전적으로 리스트 환경을 시작할 때 맨 처음 찾은 것인 숫자인지 *인지에만 좌우된다. 그 뒷항목부터는 무엇으로든 리스트임만 표시하는 것이다.

1. 우유
2. 빵
3. 치즈

그렇다면 위와 같은 상황에서 어떻게 하면 enumerate와 itemize가 차례로 오게 할 수 있을까? 다음은 한 가지 트릭이다.

```
1. 우유
1. 빵

<!--{}-->

* 치즈
* 체다
```

1. 우유
2. 빵

- 치즈
- 체다

5.7 정의 리스트

멀티마크다운에는 LaTeX의 description 환경에 해당하는 문법 구조가 없다. 그러나 이른바 *definition list*라는 것을 이 비슷하게 활용할 수는 있다. 설명이 두 줄 이상일 때 두 번째 줄부터는 **탭** 들여쓰기 하여야 한다. 빈 줄을 두어 여러 문단을 붙일 수도 있다.

치즈
: 치즈(cheese), 또는 건락(乾酪)은 우유, 산양유, 양젖 또는 그 밖의 파충류의 젖으로 만든 고체 음식이다. 원료의 종류와 제조 방법에 따라서 수백 가지가 알려져 있으며, 다양한 영양소들이 풍부하게 포함되어 있다. 또한 치즈를 만드는 대표적인 나라는 이탈리아(이태리) 등의 유럽의 나라이다.

오렌지
: 오렌지(orange)는 귤속에 속하는 과일이다. 주로 지중해같은 따뜻한 기후의 지역에서 재배된다. 껍질을 까서 생으로 먹거나, 주스 등으로 만들어 먹기도 한다.

동남아시아가 기원인 것으로 추측되며 오렌지는 이미 기원전 최고 2,500년 전에 중국에서 경작되었다. 15세기 말과 16세기 초에 이탈리아와 포르투갈 상인들이 오렌지 나무를 지중해 지역으로 들여왔다. 스페인 사람들은 1500년대 중순에 달콤한 오렌지를 미국 대륙으로 들여왔다.

치즈 치즈(cheese), 또는 건락(乾酪)은 우유, 산양유, 양젖 또는 그 밖의 파충류의 젖으로 만든 고체 음식이다. 원료의 종류와 제조 방법에 따라서 수백 가지가 알려져 있으며, 다양한 영양소들이 풍부하게 포함되어 있다. 또한 치즈를 만드는 대표적인 나라는 이탈리아(이태리) 등의 유럽의 나라이다.

오렌지 오렌지(orange)는 귤속에 속하는 과일이다. 주로 지중해같은 따뜻한 기후의 지역에서 재배된다. 껍질을 까서 생으로 먹거나, 주스 등으로 만들어 먹기도 한다.

동남아시아가 기원인 것으로 추측되며 오렌지는 이미 기원전 최고 2,500년 전에 중국에서 경작되었다. 15세기 말과 16세기 초에 이탈리아와 포르투갈 상인들이 오렌지 나무를 지중해 지역으로 들여왔다. 스페인 사람들은 1500년대 중순에 달콤한 오렌지를 미국 대륙으로 들여왔다.

5.8 각주

MMD의 각주는 정말 쉽다. 참조형 각주와 인라인 각주가 있는데, 인라인 각주는 다음과 같은 방식으로 표현한다.

```
각주[^각주란 footnote를 가리키는 말이다.]
```

각주[25]

이보다는 참조형 각주를 다음처럼 사용하는 것이 좋다.

```
각주[^fn]

[^fn]: 각주란 footnote를 가리키는 말이다.
```

각주[26]

`MMD footer` 메타데이터를 이용하여 각주만을 모아둔 별도의 mmd 파일을 자동으로 맨 마지막에 삽입되게 할 수 있다. 이 각주 참조구는 어디에 있어도 상관없으므로[27] 별도의 파일로 모아서 관리하는 것도 좋은 방법이다.

복잡한 각주, 예를 들면 동일한 각주 번호가 여러 번 출현한다든가, 표 안에 각주를 붙인다든가 하는 것은 MMD가 관여할 사항이 아니다. 그런 것이 꼭 필요하다면 최종

[25] 각주란 footnote를 가리키는 말이다.
[26] 각주란 footnote를 가리키는 말이다.
[27] 다만 다른 문장 내에 들어가면 안 되고 별도의 빈 줄을 두어서 본문과 구별되게 하여야 한다.

`.tex`을 직접 수정하거나 raw LaTeX code로 입력하도록 하라. (대부분의 경우는 그런 것이 필요하지 않다.)

5.9 수학식

MMD는 두 종류의 수학식–인라인 수식과 디스플레이 수식–을 지원한다. 인라인 수식은 `\\(`로 시작하여 `\\)`으로 끝나고 디스플레이 수식은 `\\[`로 시작하여 `\\]`로 끝난다. LaTeX의 "달러 기호"를 이용하는 수식도 지원하기는 하는데 이거야말로 문제를 일으킬 가능성이 크므로 되도록 사용하지 않는 것이 좋다(아주 짧은 행중 수식에 `$n$` 정도를 표기하는 경우가 아니라면).

```
피타고라스의 정리에 의하면 \\(a^2+b^2=c^2\\)이므로
```

> 피타고라스의 정리에 의하면 $a^2 + b^2 = c^2$ 이므로

```
이차방정식 \\(ax^2+bx+c=0\\)의 근의 공식은
\\[
x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}
\\]
이다.
```

> 이차방정식 $ax^2 + bx + c = 0$ 의 근의 공식은
>
> $$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$
>
> 이다.

수식은 Marked 2 앱에서 MathJax를 활성화하면 입력하는 즉시 볼 수 있다.

수식의 번호

수식에 번호를 붙여야 할 때가 있다. `mmd-oblivoir`는 `\autotag`라는 명령을 하나 제공한다.

```
\\[
a^2+b^2=c^2 \autotag
\\]
```

$$a^2 + b^2 = c^2 \tag{1}$$

이것은 미리보기에서는 보이지 않지만 나중에 LaTeX으로 변환하여 컴파일하면 제대로 된 모양으로 나타난다.

수식 번호와 참조

수식 내부에서는 LaTeX 명령이 그대로 듣기 때문에 `\label`은 다음과 같이 달면 된다.

```
\\[
a^2+b^2=c^2 \autotag \label{eq:pt}
\\]
```

$$a^2 + b^2 = c^2 \tag{2}$$

이것을 참조하려고 [eq:pt][]라고 하면 잘 되지 않는다. 그 이유는 멀티마크다운이 수식 모드 안에 있는 것들에 대해 별다른 관여를 하지 않기 때문이다. 그러므로 수식 참조에 한해서 `<!--\eqref{eq:pt}-->`와 같은 문법을 쓰자. 식 (2)를 참고하라.

여러 줄 수식

여러 줄 수식은 amsmath에서 정의하는 변형 환경들, 예를 들면 독립된 환경인 `align`이 아니라 math mode에서 동작하는 `aligned`를 쓰는 방식으로 코딩하는 것이 좋다. 만약 `align`을 반드시 써야 한다면 HTML 주석문으로 escape하여야 한다.

```
<!--\begin{align}
X &= a + b \\
  &= c + d
\end{align}-->
```

그런데 이렇게 하면 수식 미리보기가 잘 안 된다.

```
\\[
\begin{aligned}
X &= a + b \\
  & = c + d
\end{aligned} \autotag
\\]
```

$$\begin{aligned} X &= a + b \\ &= c + d \end{aligned} \tag{3}$$

5.10 HTML 주석문과 라텍 코드

MMD는 HTML 주석문 안에 있는 내용을 LaTeX 파일로 변환할 때 수정 없이 넘겨준다. MMD가 전혀 지원하지 않는 문단 구조인 theorem-류 문단들은 이를 이용하여 코딩한다.

```
<!--\begin{thm}--> [피타고라스의 정리]
직각 삼각형의 세 변 사이에 다음 관계가 성립한다.
\\[
a^2+b^2=c^2
\\]
<!--\end{thm}-->
```

정리 5.1 (피타고라스의 정리) 직각 삼각형의 세 변 사이에 다음 관계가 성립한다.

$$a^2 + b^2 = c^2$$

`mmd-oblivoir`에 `thm` 환경이 이미 정의되어 있다. 자세한 사항은 `mmd-oblivoir-*.tex`과 함께 오는 `progress-thm.tex`을 참고하라. 이 샘플 파일은 `thm`과 `proof` 환경만을 정의하고 있다.

HTML 주석문은 이밖에도 `abstract`나 `\tableofcontents` 명령 등을 지시하기 위해 사용할 수 있다. 이 기능들은 `mmd-oblivoir`에는 빠져 있으므로 필요하면 적당한 곳에 HTML 주석문 형식으로 적어넣으면 된다.

상호 참조 (subsection 6.4)와 관련하여 한글 자동조사 명령은 어쩔 수 없이 HTML 주석문 안에 넣어야 한다.

HTML 주석은 대단히 강력한 LaTeX 명령 입력을 가능하게 한다. LaTeX만을 생각한다면 이것으로 무엇이든 할 수 있다. 그러나 `.mmd`를 만드는 목적이 HTML이나 OpenDocument까지 고려하고 있다면 이것은 하나의 장애가 된다. 즉 HTML 주석으로 삽입한 코드는 HTML로 출력할 때는 완전히 무시될 수밖에 없다는 것이다.

5.11 주석 달기

HTML 주석문이 LaTeX 날코딩을 넘겨주기 위해 사용되므로 진짜 주석(comment)를 어떻게 기록해야 할지 알 수 없게 되었다. HTML이 출력 포맷이라면 이른바 CriticMarkup[28]을 이용하는 방법이 있겠지만 LaTeX을 백엔드로 쓰는 상황에서는 그다지 도움이 되지 않는다. 역시 이런 경우도 HTML 주석문을 쓰는 것이 가장 확실한 방법일 것이다.

```
<!--
% 이 부분이 주석문입니다.
-->
```

[28] CriticMarkup에 대해서는 MultiMarkdown 사용자 설명서를 참고하라.

스크리브너를 사용하고 있다면 그 앱이 제공하는 다양한 note 기능을 이용할 수 있다. 스크리브너 내에서는 보이지만 export(스크리브너의 용어로는 "컴파일")할 때 반영되지 않는 footnote와는 구별되는 것이 있으므로 그것을 이용하여 주석문을 달면 된다.

5.12 문단 나누기

MMD 소스의 행나눔은 LaTeX 파일에 "있는 그대로" 출력된다. 그러므로 LaTeX을 백엔드로 사용하려 하는 MMD에서 문단 나누기는 반드시 빈 줄 하나를 두어야 한다.

문단과 문단 사이에 임의로 공행을 하나 두고 싶을 때는 어떻게 할 것인가? 일반적으로 모든 문단 사이에 공백 행을 두고자 한다면 LaTeX 헤더 파일 설정에서 `\parskip` 값을 지정하는 것이 좋다. 그렇지 않고 특별히 빈 줄을 두어야 하는 경우라면 HTML 주석문을 이용하라.

문단

```
<!--\bigskip-->
```

다음 문단

`\fancybreak`를 넣고자 할 때도 마찬가지로 한다. 즉, 이것은 MMD 마크업이 담당할 부분이 아니다.

6 그밖의 멀티마크다운 문법

6.1 그림

MMD는 그림을 두 가지 방식으로 처리한다.

1. 인라인 그림. `\includegraphics`로 해당 그림을 현재 위치에 찍는다.
2. figure 그림. 멀티마크다운이 그림을 figure 환경 안에 넣어준다.

인라인 그림이냐 figure 그림이냐는 그림에 대한 마크업이 있기 전에 빈 줄이 있느냐 없느냐에 달린 것이다. 빈 줄이 없으면 이전 문단 안에 들어가는 인라인 그림으로 간주하고 아래 위로 빈 줄이 있어서 그림 마크업이 독립 문단으로 되어 있으면 figure 그림이 된다.

구체적인 예를 들어 보자. 샘플 그림은 mwe 패키지가 제공하는 `example-image.png`로 하기로 한다.

```
여기에 인라인 이미지 ![image][imgid]

[imgid]: example-image.png height=12pt
```

여기에 인라인 이미지 

한편, figure 그림은 그 자체가 별개의 한 문단을 이루고 있어야 한다.

```
![테스트 그림][imgtest]

[imgtest]: example-image.png width=200px
```

이 경우 [테스트 그림] 부분이 \caption으로 써진다. 그리고 label은 "imgtest"가 된다.

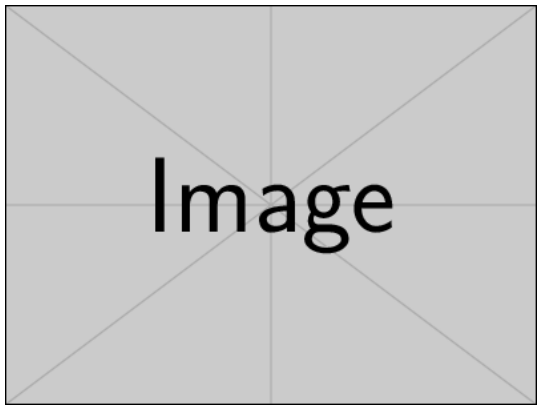


그림 2: 테스트 그림

그림에 대한 것은 이 정도이다. 이보다 더 세밀한 LaTeX으로는 할 수 있는 온갖 기능들은 LaTeX으로 하면 된다. 예컨대 subfigure같은 기능은 마지막 단계에서 tex 소스를 수정하는 것이 옳다.

6.2 URL 링크

이것이야말로 MMD가 LaTeX보다는 HTML을 겨냥한 문서작성도구라는 것을 잘 보여주는 것이다. URL 링크를 걸 수 있는 멀티마크다운의 다음 몇 가지 문법을 보자.

```
<http://ktug.org>

[KTUG](http://www.ktug.org)

[Korean TeX Society][KTS]

[KTS]: http://www.ktug.org
```

http://ktug.org

KTUG[29]

Korean TeX Society[30]

이들은 각각 다음과 같이 변경된다.

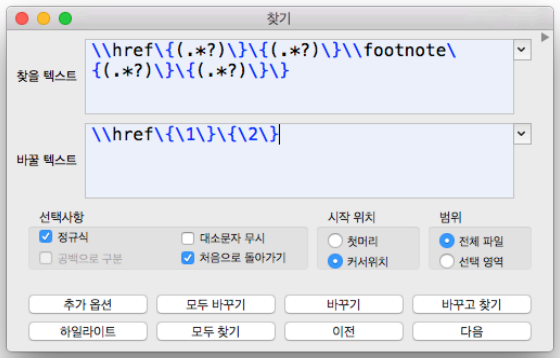
```
\href{http://ktug.org}{http:/\slash ktug.org}

\href{http://www.ktug.org}{KTUG}
     \footnote{\href{http://www.ktug.org}{http:/\slash www.ktug.org}}

\href{http://www.ktug.org}{Korean TeX Society}
    \footnote{\href{http://www.ktug.org}{http:/\slash www.ktug.org}}
```

무슨 일이 벌어졌는지 알 수 있을 것이다. URL 링크는 LaTeX에서 `\href`으로 번역되고 첫 번째 경우처럼 url만 준 경우를 제외하면 모두 각주가 붙는다.

URL 링크에 각주를 붙일 것이냐 말 것이냐 문제는 전적으로 취향의 문제이다. 대체로 나는 각주가 있는 편이 좋다고 생각하는데 이것이 번거롭다면 마지막 단계에서 `.tex`으로 바꾼 후에 이 각주를 regular expression replace를 이용하여 간단히 바꾸면 된다. TeXshop의 경우 다음과 같이 하면 pdf에서 링크는 남고 각주는 없어진다.



6.3 표

table은 참 어려운 문제이다. LaTeX에서도 table (tabular)를 제대로 구현하는 데 많은 노력을 기울여야 한다. 최근 `tabu` 패키지가 나와서 표를 원하는 대로 그리는 데 많은 진전을 이루었으나, MMD 자체는 아직 tabu를 지원하지 않는다. 그 대신 MMD 문법이 제공하는

[29] http://www.ktug.org

[30] http://www.ktug.org

표는 무조건 tabulary 패키지를 이용하는 표로 변환된다. 그리고 표의 기본적인 모양은 booktabs 표이다.

다음은 기본적인 표 그리기 코드이다.

```
|              |          Grouping           ||
First Header   | Second Header | Third Header |
 ------------- | :-----------: | -----------: |
Content        |          *Long Cell*        ||
Content        |   **Cell**    |         Cell |

New section    |     More      |         Data |
And more       | With an escaped '\|'         ||
[Prototype table]
```

이 코드에 대한 자세한 설명이 MultiMarkdown 사용자 설명서에 있으므로 해당 부분을 참고하라. 이 표는 Prototype table (Table 1)과 같이 표현된다.

표 1: Prototype table

	Grouping	
First Header	Second Header	Third Header
Content	*Long Cell*	
Content	**Cell**	Cell
New section	More	Data
And more	With an escaped '|'	

최근 KTUG[31]의 한 대화에서 longtable을 MMD와 LaTeX에서 동시에 구현할 수 없겠느냐는 질문이 있었다. http://www.ktug.org/xe/index.php?document_srl=188967. 여기에 대해 내가 제시한 해결책이 스크리브너와 longtable[32]이라는 글이다. 이 샘플을 보면 어떤 방식으로 표 모양을 제어할 수 있을지 짐작할 수 있을 것이다. 필요하다면 tabu의 표로 변형하는 것도 불가능하지는 않아 보인다. 그러나 어쨌든 이런 부분은 LaTeX 코딩의 문제이지 MMD의 문제는 아니다.

6.4 상호 참조

LaTeX을 쓰는 즐거움 중의 하나가 label-ref이다. 대략 다음과 같은 개체에 label을 붙인 다음 그 참조자로 \ref하여 상호 참조(cross-reference)한다.

- 장절 표제

[31] http://www.ktug.org/

[32] http://doeun.blogspot.kr/2014/11/scrivener-longtable.html

- 그림
- 표
- 수식

여기서 더 나아가 `\item`에 대해서 label-ref하는 경우도 없지 않으며 사용자가 정의한 환경에서 label-ref가 가능한 경우도 있다. 이런 비범용적인 상황을 제외하면 대략 위에 언급한 요소들이 상호참조의 대상이 된다.

MMD에서 이들에 대한 참조 방법을 알아보자.

장절 표제로의 참조

장절 표제로 참조하는 것은 매우 쉽다. 예컨대 장절이 다음과 같이 코딩되어 있다면

```
# 장 표제 #
```

`[장 표제][]`와 같이 적어주는 것으로 참조가 가능해진다. 이것은 `\autoref`에 의한 것이므로 한글 문서에 반드시 적합하다고 할 수는 없지만 아무튼 상호 참조는 이루어진다. 예를 들어 `[상호 참조][]`와 같이 코딩하면 "상호 참조 (subsection 6.4)"로 링크가 붙는다.

`\autoref`의 동작을 재정의하여 원하는 모양으로 출력할 수 있으나 이것 역시 LaTeX 프로그래밍 문제이므로 여기서 더 논의하지 않겠다.

표로의 상호 참조

MMD로 그린 표라면 caption을 붙인 경우 표로의 참조가 가능하다. 예를 들어 Prototype table (Table 1)에 대해서 참조가 이루어진다. 앞선 절(표 (subsection 6.3))에서 "Prototype table"이라는 이름의 표를 작성한 바 있으므로 다음과 같이 코딩하면 된다.

```
[Prototype table][]
```

이 역시 `\autoref`을 이용하는 것이다.

그림으로의 참조

그림으로의 참조는 조금 생각할 것이 있다. 원래 MMD가 HTML을 주타겟으로 하는 것임을 기억하자. 그림~`\ref{fig:test}`와 같은 방식의 상호 참조는 LaTeX에 고유한 것이다.

예컨대 다음과 같은 방식으로 figure 그림을 삽입한 경우라고 하자(인라인 그림은 ref 될 수 없다).

```
![This is Image Caption][figureid]

[figureid]: example-image.png width=100px
```

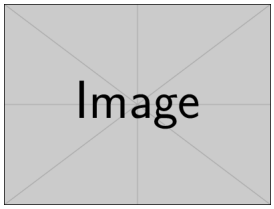


그림 3: This is Image Caption

이 경우에 참조자는 `figureid`이다. 이것을 일반적인 상호 참조 방식으로

```
[figureid] []
```

이렇게 쓰면 그 결과는 다음과 같다.

```
\href{example-image.png}{figureid}
  \footnote{\href{example-image.png}{example-image.png}}
```

즉, figureid[33]와 같이 나타난다는 것이다.

이것으로 충분하다면 더 언급할 것이 없다. 다만 LaTeX 고유의 방식으로 참조하고자 한다면 LaTeX 자체의 언어를 쓰면 된다.

```
<!--\fref{figureid}\를--> 참조하시오.
```

그림 3을 참조하시오.

수식으로의 참조

수식은 완전히 별개의 문제이다. 그러므로 수식 번호와 참조 (section 5.9)에서 말한 대로 HTML 주석문 안에 참조 명령을 넣는 것이 가장 쉬워 보인다.

6.5 용어집

이 절에서 설명하는 내용은 `mmd-oblivoir`에 바로 정의되어 있지 않은 것이다. 이 기능들이 필요하다면 자신의 `preamble`을 만들어서 아래 설명하는 대로 설정을 추가하여 사용하도록 하라. 자신의 preamble을 추가하는 방법은 아래 소절 "메타데이터와 header 파일 만들기 (section 7)"를 참고한다.

[33] example-image.png

glossaries

MMD에서 다음과 같은 문법으로 작성된 "용어 각주(glossary footnote)"는 glossaries 패키지를 이용하여 용어집 변환을 할 수 있게 한다.

```
용어[^glossarytest]에 대해 알아보자.

[^glossarytest]: glossary: 용어집
    이것은 용어를 지칭한다
```

LaTeX 입장에서 용어집을 만들려면 다음과 같은 조건이 필요하다.

1. preamble에 `\usepackage{glossaries}`가 선언되어야 한다.
2. preamble에 `\makeglossaries` 명령이 놓여야 한다.
3. 용어집이 놓일 위치에 `\printglossaries` 명령이 있어야 한다.
4. 처음 컴파일한 후에 명령행에서 `makeglossaries filename` 명령이 한 번 실행되어야 한다.

`mmd-oblivoir`는 이 과정을 사용자가 알아서 하도록 하고 있다. 즉, 정의되어 있지 않다. 그러므로 `myglossaries.tex` 파일을 다음 내용으로 제작하여 `latex input`하도록 해야 한다.

```
\usepackage{glossaries}
\makeglossaries
```

그리고 (만약 용어집이 문서의 끝에 온다면) `myglossariesfooter.tex`을 다음 내용으로 제작하여 `latex footer`하도록 하여야 한다.

```
\printglossaries
```

때에 따라 `\printglossaries`는 문서의 처음에 와야 할 수도 있다. 이 때는 HTML 주석문을 이용하라.

한 번 컴파일한 후에 `makeglossaries filename` 명령을 실행해야 용어집이 만들어진다는 사실을 기억하자. 참고로 이 과정은 MultiMarkdown 사용자 설명서에 설명되어 있는 것과는 조금 다르지만 현재 적용 가능한 것이므로 이 지침을 따르는 것이 좋다.

acronym

약어(acronym)을 쓰는 규약은 다음과 같다.

```
MMD

*[MMD]: MultiMarkdown
```

약어 정의는 하나의 파일에 모아서 `mmd input`하는 것이 바람직하다. 그리고 LaTeX에서 이 약어에 해당하는 것들은 모두 `acronym` 패키지의 `\ac` 명령으로 치환된다. `mmd-oblivoir`는 acronym 패키지를 로드하는 부분이 없으므로 직접 다음 내용을 `mypreamble.tex`에 적어 넣고 이 파일을 `latex input`한다.

```
\usepackage{acronym}
```

약어 정의의 편리한 점은 일일이 `[MMD]`와 같이 코딩하지 않아도 `MMD`에 해당하는 부분은 모두 `\ac` 처리를 해준다는 점이다. 다만 한글 문자는 약어로 쓸 수 없다. 약어로 사용가능한 것은 원칙적으로 대문자 ASCII이다. 자세한 사항은 MultiMarkdown 사용자 설명서를 보라.

6.6 문헌목록과 인용

인용에 사용되는 멀티마크다운 신택스는 `[][#참조자]` 형식이다.

```
이 부분은 [][#Talbot2008]에서 찾아볼 수 있다. 또는 [][#Talbot2008;]을
보라.

[#Talbot2008]: N. Talbot. "Creating a LaTeX Minimal Example", Dickimaw.
```

이 부분은 [3]에서 찾아볼 수 있다. 또는 [3]을 보라.

이 예제에서 앞의 것 `[][#Talbot2008]`은 `\citep`가 되고 뒤의 것 `[][#Talbot2008;]`은 `\citet`로 된다. 그리고 처음 나온 `[]`에는 인용 위치를 적을 수 있다. 예: `[p. 25][#Talbot2008;]`.[34] 참고로 현재 버전의 MMD를 이용하면 `\citep`의 경우 `~\citep`와 같이 공백 하나를 앞에 둔다. `\citet`는 그렇지 않다. 영어식 인용에서는 이것이 편리하지만 한글 문헌에서는 이 공백이 보기 싫을 수도 있다. 그것은 최종적으로 수정할 때 반영하도록 하자.

이 인용은 문헌 목록으로부터 이루어지는데 문헌 목록을 관리하는 방법은 두 가지가 있다. 그 두 가지란, LaTeX 식으로 말하자면 `thebibliography` 환경을 직접 입력할 것인지 BibTeX 데이터베이스를 활용할 것인지에 대응한다.

문헌목록을 mmd로 유지

마치 LaTeX에서 `thebibliography` 환경을 담고 있는, 이를테면 `ref.tex`을 만들어서 관리하듯이, MMD에서도 그렇게 할 수 있다. 비교적 짧은 글에서는 각주 붙이듯이 문헌 목록

[34] natbib 패키지가 명시적으로 로드되지 않으면 `\citet`와 `\citep`가 사실상 동일하게 동작한다. `mmd-oblivoir`가 이렇게 초기 설정되어 있다. 그러므로 다른 추가 설정 없이 `mmd-oblivoir`에 문서 내 문헌 목록을 쓰는 경우 `\citet`로의 변환이 이루어지지 않을 수 있으므로 `\citep` 형식만 쓰도록 하라.

도 인용 표지가 있는 문단 바로 밑에 적어두어도 별로 불편할 게 없지만 긴 글의 경우는 문헌 목록만 별도로 관리하는 것이 더 합리적이다.

예를 들면 다음과 같은 내용으로 된 문헌 목록 MMD를 작성한다.

```
[#sample1]: A. U. Thor, *A Book for the Christmas*, Unknown Press, 2014.

[#sample2]: John Doe, *A Real Tester's Testing*, Testers Publisher, 2014.
```

이 `ref.mmd` 파일을 작업 디렉터리에 두고 작업 (메인) 파일의 메타데이터에 다음 사항을 기록한다.

```
MMD footer: ref.mmd
```

이제 본문에서 이 파일에 있는 참조자를 이용하여 인용하면 된다. MMD는 `ref.mmd`에 있는 것 중에서 "인용된 것"만 LaTeX의 `thebibliography` 환경으로 만든다. 만약 인용되지 않은 것도 포함시키고 싶다면 LaTeX의 `\nocite`에 해당하는 다음 문장을 포함하여야 한다.

```
[Not Cited][#example2]
```

이 방식으로 만들어지는 LaTeX `thebibliography`는 "인용순"으로 되어 있다. MMD가 인용된 문헌을 알파벳순으로 바꾼다든지 하는 일은 해주지 않는다. 이 문제 해결을 위한 작은 유틸리티를 만든다는 이야기를 듣기도 하였지만 시험해보지 않았고, 만약 정말로 문헌 목록의 정렬이 문제가 된다면 (큰 문서라면) BibTeX 같은 다른 방법을 생각해보는 것이 더 낫다. 작은 문서이고 문헌 목록이 그리 길지 않다면 최종 `.tex` 파일에 대하여 수작업해도 별 문제 없을 것이다. 선택한 행을 abc 순으로 정렬해주는 에디터도 많다.

BibTeX을 써보자

BibTeX은 대단히 강력한 문헌 데이터베이스 관리 도구이다. 작업 흐름에서 `bibtex`이라는 외부 프로그램을 한 번 실행해주어야 한다는 번거로움이 있기는 하지만 이것은 LaTeX workflow utility를 사용하면 대부분 자동으로 처리된다. 방대한 문헌을 포함하는 문서를 작성한다면 이 방식의 접근이 시간을 절약해줄 가능성이 많다. 그리고 최근에는 유니코드를 더 잘 지원하는 `biber`가 각광받고 있다. 아쉬운 것은 한글 문헌의 처리 방법이 완전히 깔끔하지는 않다는 것 정도이다. [4]가 이 문제에 대한 한 가능성을 제시하고 있으므로 참고하라.

BibTeX의 데이터베이스 파일 `*.bib`를 직접 편집하는 것은 꽤 고단하다. 이를 쉽게 관리하게 해주는 툴들이 많은데 그 중에서 (맥에서만 된다는 문제가 있기는 하지만)

BibDesk[35]를 추천한다. MacTeX[36]을 설치하면 함께 설치된다. 이밖에도 JabRef[37] 같은 훌륭한 bib 에디터들이 몇 있다.

먼저 bib 파일을 편집하기로 하자. 다음과 같은 test.bib을 생각해본다. 한 개 항목만 넣었다.

```
@booklet{karnes2014,
    url = {http://wiki.ktug.org/wiki/wiki.php/KTUGExtDocArchive},
    Author = {Nova~De~Hi},
    Howpublished = {on-line 문서},
    Title = {스크리브너와 라텍},
    Year = {2014}}
```

이 문헌으로부터의 인용은 본문에서 `[][#karnes2014]`라고 하는 것으로 충분하다. 문제는 문헌 목록을 만들어 넣는 것.

MMD가 상정하는 바 문헌 목록을 이용하는 표준적인 LaTeX 코드는 다음과 같다. 잘 보면 natbib이 기본적으로 사용되고 있음을 알 수 있다.

```
\documentclass{article}
\usepackage[authoryear]{natbib}

\begin{document}

\citet{karnes2014}

\bibliographystyle{plainnat}
\bibliography{test}
\end{document}
```

이 가운데 MMD 메타데이터로 조작할 수 있는 부분은 \bibliographystyle과 \bibliography이다. 각각 biblio style과 bibtex이 여기에 해당한다. plainnat는 바꾸지 않아도 되므로 우리는 .bib 파일의 이름만 다음과 같이 하여 넘겨주자.

```
bibtex: test
```

여기서 확장자 .bib는 쓰지 않는다.

이제 문서의 마지막에 문헌 목록을 붙여야 한다. 이를 위하여 myfooter.tex을 작성하고 이것을 latex footer 메타데이터로 불러들이도록 설정하자. 메타데이터는 다음과 같은 모양이 된다.

[35] http://bibdesk.sourceforge.net/

[36] https://tug.org/mactex/

[37] http://jabref.sourceforge.net/

```
...
latex footer: myfooter
latex input: mmd-oblivoir-begin-doc
```

`myfooter.tex` 파일은 다음 내용으로 작성한다.

```
\ifx\bibliostyle\undefined
    \bibliographystyle{plainnat}
\else
    \bibliographystyle{\bibliostyle}
\fi

\bibliocommand
```

이상을 정리하면, MMD 파일은 이렇게 된다.

```
latex input: mmd-oblivoir-header
base header level: 3
title: BibTeX 테스트
author: John Doe
bibtex: test
latex footer: myfooter
latex input: mmd-oblivoir-begin-doc

# 문헌 인용 테스트

[][#karnes2014;]
```

마지막으로 `mmd2tex`한 결과에 대하여 `xelatex -> bibtex -> xelatex -> xelatex`으로 컴파일하면 문헌 목록과 인용이 모두 잘 나타난다.

bibLaTeX을 써보자

전통의 BibTeX이 아니라 요즘 뜨는 biblatex과 biber를 이용하는 방법을 생각해보자. 이렇게 하기로 했다면 `biblio style`이나 `bibtex` 메타데이터가 의미 없다. `bib` 파일은 앞선 절에서 만든 `test.bib`을 그대로 써본다.

biblatex을 이용하려면 `mypreamble.tex`에 다음 내용을 넣어야 한다.

```
\ifx\citet\undefined\else \let\citet=\relax\fi
\ifx\citep\undefined\else \let\citep=\relax\fi
\usepackage[natbib=true,backend=biber]{biblatex}
\addbibresource{\mybiblatexresource}
```

그리고 문서의 마지막에 문헌 목록을 넣기 위하여 myfooter.tex에 다음 내용을 추가한다.

```
\printbibliography
```

이제 작업 MMD 파일의 메타데이터를 다음과 같이 설계한다.

```
latex input: mmd-oblivoir-header
...
my biblatex resource: test.bib
latex input: mypreamble
latex footer: myfooter
latex input: mmd-oblivoir-begin-doc
```

my biblatex resource라는 메타데이터를 추가하였다. 이 "비표준" 메타데이터는 멀티마크다운이 \def\mybiblatexresource{test.bib}으로 넘겨주기 때문에 mypreamble보다 먼저 나와야 한다.[38]

본문에서 [][#karnes2004]와 같은 인용 명령이 한 번 이상 사용되었다면 최종적으로 컴파일할 때 xelatex -> biber -> xelatex 순으로 처리하면 문헌 목록이 예쁘게 들어간다. 소팅 순서 등은 biblatex의 옵션으로 지정할 수 있다. 그밖에 문헌 목록의 명칭을 "참고문헌"으로 바꾼다든가, 한글 문헌에 맞게 설정을 추가한다든가 하는 것은 전적으로 LaTeX 영역의 일이며 mypreamble.tex을 적절하게 수정하면 된다.

참고로 위에 보인 방법으로는 my biblatex resource에 bib 파일을 하나밖에 지정할 수 없다. 여러 개의 bib 파일을 쓰도록 세련되게 고치고 싶으니 다음과 같이 해보자. 약간 고급 팁이 될라나?

먼저 메인 파일의 메타데이터는 다음과 같은 모양이 되게 한다. 즉, ;로 각 항목을 구분하는 것이다. 이 메타데이터가 latex input: mypreamble보다 먼저 나와야 한다.

```
my biblatex resource: test.bib; test2.bib
```

그리고 mypreamble.tex에 정의를 추가한다.

```
\let\citet=\relax
\let\citep=\relax
\usepackage[natbib=true,backend=biber]{biblatex}
\ExplSyntaxOn
\DeclareDocumentCommand \AddBibResource
  { > { \SplitList { ; } } m }
  { \ProcessList {#1} { \addbibresource } }
```

[38] 만약 이 항목이 정의되어 있지 않으면 \jobname.bib을 쓰라고 하거나 에러가 발생하지 않도록 정의하는 것은 LaTeX 쪽에서 하면 될 터이다. 여기서 이 문제는 더 다루지 않겠다.

```
\ifx\mybiblatexresource\undefined \else
\exp_args:Nx \AddBibResource{\mybiblatexresource}
\fi
\ExplSyntaxOff
```

이제 둘 이상의 bib 파일도 사용할 수 있게 되었다.

6.7 프로젝트 관리

단행본 원고나 학위논문 같이 꽤 길고 많은 장(chapter)으로 이루어진 문서를 작성할 때, 각 장별로 별도의 MMD 파일로 작성하는 것이 좋다. 기본적으로 글쓰기는 앞에서부터 순차적으로 이루어지는 것이 아니고 나중에 나올 내용을 먼저 쓰게 되기도 하며 때에 따라 순서를 뒤섞기도 해야 하기 때문에 하나의 MMD 파일에 모든 내용을 전부 넣는 것은 이럴 경우 불편을 초래할 수 있다. 이것을 LaTeX 세계에서는 "프로젝트 관리"라고 부르는데 `\include` 명령을 잘 활용하여 하나의 메인 파일과 부속 파일로 구성한다.

비슷한 것을 멀티마크다운에서도 할 수 있다. 예컨대 메인 파일이 `main.mmd`이고 제1장이 `chapter1.mmd`라면 `main.mmd`에서 다음과 같이 마크업하면 해당 파일의 내용을 전부 불러들인다.

```
{{chapter1.mmd}}
```

이 기능을 "파일 포함하기(File Transclusion)"라고 한다. 각각의 장별 파일을 작업할 때는 그 MMD에 기록된 메타데이터에 따라 작업 문서가 만들어지지만 상위 메인 파일에 이것이 포함되면 부속 파일의 메타데이터는 "무시되고" 메인 파일의 메타데이터만 의미를 갖는다. 요컨대, 장별로 작업할 때도 그 자체로 하나의 문서가 되고 전체를 하나의 문서로 모아서 출력할 수도 있다는 것이다.

원한다면 부속 파일을 별도의 폴더에 모아놓을 수도 있다. 이 때는 메인 파일에 다음과 같은 메타데이터를 기록한다.

```
transclude base: ./chapters/
```

이 값이 없으면 같은 폴더에서 포함할 파일을 찾는다.

7 메타데이터와 header 파일 만들기

7.1 메타데이터

LaTeX을 백엔드로 하는 MMD 문서 작성에서 알아두면 좋을 것 같은 "표준" 메타데이터를 정리해본다.

latex input 여기에 해당하는 값을 `\input`하도록 한다. 여러 번 반복 사용할 수 있으며 지정된 순서대로 파일을 로드한다.

latex footer MMD가 마지막에 `\end{document}`를 적어넣기 전에 `\input`할 파일을 지정한다.

latex mode 여기서는 `beamer` 문서를 만들려면 필요하다는 정도로 언급해두겠다. 실제로 MMD는 `memoir.xslt`를 기본으로 동작하므로 beamer 문서를 만들려면 조금 다른 방식의 동작이 필요하고 그것을 지정하는 것이다. 추가적인 XSLT를 사용하지 않는 경우에 여기 올 수 있는 값은 latex, memoir, beamer 셋이고 memoir가 기본이다. 당연히 `mmd-oblivoir`는 memoir.xslt에서 동작한다.

latex xslt 필요하다면 자신이 작성한 xslt를 이용하도록 할 수 있다. 이 값을 활성화하면 `multimarkdown` 명령을 직접 (명령행 옵션을 주어서) 쓰거나 `mmd2tex-xslt` 스크립트를 이용하여야 한다.

latex title 그냥 `title`과 같다. 표제지를 만드는 데 쓰인다.

latex author 그냥 `author`와 같다. 표제지를 만드는 데 쓰인다.

biblio style BibTEX 모드로 동작할 때 MMD의 기본 스타일은 natbib이다. natbib이 아닌 다른 스타일을 쓰려면 이 값을 준다. (보통은 필요없다.)

bibtex 포함할 `.bib` 파일을 확장자 없이 적는다.

mmd footer `latex footer`가 `.tex` 파일을 마지막에 불러들이는 것인데 비해, 이 메타데이터는 현재 작성 중인 `.mmd` 파일의 마지막에 원하는 파일을 포함하게 하는 것이다.

base header level base header level에 대해서는 앞서 장절 표제 (subsection 5.1) 절에서 설명하였다.

date 표제에 적을 날짜를 넘겨준다. 이 메타데이터가 없으면 오늘 날짜를 찍는다.[39]

더 자세한 사항은 MultiMarkdown 사용자 설명서를 참고하라.

기억해두어야 할 것은 메타데이터의 value에 오는 텍스트는 멀티마크다운 마크업이 안 된다는 것이다. 즉 HTML 주석문도 쓸 수 없고 강조를 위한 * 신택스도 쓸 수 없다. (단, LaTeX의 유보 문자들은 변환된다.) 이 점을 잘 기억하여 문서 작성에 활용하여야 한다. 이 위치에 올 텍스트가 복잡한 것이라면 되도록 단순하게 해두었다가 나중에 수정하는 방식으로 접근하는 것이 편하다.

[39] `mmd-oblivoir`에서 `date` 메타데이터를 이용하여 `\maketitle`하도록 2014/12/05일 버전에서 수정되었다. 그 이전 판에서는 항상 `\today`로 찍혔다.

7.2 헤더 파일의 구조

표준적인 LaTeX 파일은 다음처럼 되어 있다.

```
\documentclass{oblivoir}

\usepackage{jiwonlipsum}

\title{Title}
\author{Author}
\date{Date}

\begin{document}

\maketitle

\tableofcontents

\section{First Section}

\jiwon

\end{document}
```

이를테면 MMD는 `\section{First Section}`부터 코딩하는 셈이다. 따라서 그 이전의 모든 설정들을 헤더 파일에 넣어두고 그것을 `\input`하는 방식으로 접근하는 것임을 알 수 있다.

그렇다면 왜 한 개의 헤더 파일만 부르지 않고 `*-header.tex`과 `*-begin-doc.tex`으로 둘로 나누어 두었는가? 그 이유는 preamble을 MMD 마크업(특히 메타데이터)으로 제어하게 하기 위해서이다. 복잡한 preamble 설정을 가능하게 하고 이것을 의도대로 수정하게 하려는 데 이 방식이 적합했다.

`*-header.tex`에는 `\documentclass` 문이 들어 있다. 그리고 `*-begin-doc.tex`에는 `\begin{document}`가 들어 있다. 그러므로 `*-begin-doc`이 마지막으로 `latex input` 되어야 한다. `\maketitle`이나 `\tableofcontents`같은 것도 `*-begin-doc`에 정의되는 것이다.

만약 자신만의 명령이나 환경을 정의하거나 레이아웃, 폰트 등을 조절하고 싶다면 `*-header.tex` 파일을 수정하거나, 아니면 자신만의 정의를 모아서 `mypreamble.tex`을 작성하고 다음과 같은 순서로 `latex input`을 표시하면 되는 것이다.

```
latex input: mmd-oblivoir-header
latex input: mypreamble
latex input: mmd-oblivoir-begin-doc
```

MMD는 `\end{document}`를 스스로 만든다. 그런데 `\printindex`나 bibliography 등을 배치하는 명령을 정의한 `myfooter.tex`이 있다면 이것을

```
latex footer: myfooter
```

위와 같이 정의하여 활성화할 수 있다.

이 글에서 제공하는 `mmd-oblivoir-header`, `mmd-oblivoir-begin-doc`은 하나의 예제이다. 자신만의 스타일로 출력물을 만들고 싶다면 위의 지침을 잘 따라서 스스로 헤더 파일들을 만들 수 있을 것이라고 생각한다.

8 찾아보기 만들기

LaTeX을 쓰는 즐거움 중의 하나가 찾아보기(Index)를 상당히 쉽게 생성할 수 있다는 것이다. 물론 `\index` 엔트리를 소스에 코딩해 넣어야 하는 것이 만만한 일은 아니다. 이따금 소스 코드의 특정 단어를 "노려보면" 컴퓨터가 자동으로 해당 단어에 `\index{단어}`를 추가해주는 인공지능은 언제나 나올려나 쓸데없는 생각을 하게 만들기도 한다.

MMD는 이 index를 지원하는 마크업을 바로 제공하지 않는다. 따라서 트릭이 필요하다. 여기서는 `makeindex`가 아니라 `texindy`를 사용한다고 가정하겠다. 유니코드니까.

8.1 치환을 이용하자

가장 손쉽게 할 수 있는 것은 MMD 마크업에 "표지"를 달아두었다가 그것을 최종 `.tex`에서 일괄 치환하는 방법이다. 물론 정규식(Regex)를 이용해서.

일단 `mypreamble.tex`에 다음과 같은 정의를 하나 추가하기로 한다.

```
\ExplSyntaxOn
\DeclareDocumentCommand \WordIndex { o m o }
  {
      #2
       \IfNoValueTF { #1 }
       {
           \index{#2}
       }
       {
           \index{#1!#2}
       }
       \IfNoValueTF { #3 }
       {
           \index{#2}
       }
       {
```

```
            \index{#2@#3}
        }
    }
\ExplSyntaxOff
\makeindex
```

이 명령의 의미는 예컨대 `\WordIndex{단어}`라고 하면 단어`\index{단어}`로 변환하라는 것이다. 옵션이 앞에 오면 `\WordIndex[상위단어]{단어}`를 단어`\index{상위단어!단어}`로 바꾸고 만약 옵션이 뒤에 오면 `\WordIndex{단어}[위치]`를 단어`\index{단어@위치}`로 만든다. 원한다면 더 복잡한 명령도 만들 수 있겠지만 일단 이 정도.

MMD에서 예컨대 다음과 같이 코딩하였다고 하자.

```
=단어=
```

이것은 `.tex`으로 `mmd2tex`하면 그대로 `=단어=`로 넘어온다. TEXshop의 정규식 바꾸기를 이용하여 이것을 모두 `\WordIndex{단어}`로 바꾸어주면 된다.

이제 다음 한 줄로 되어 있는 footer 파일을 작성하자(`myfooter.tex`).

```
\printindex
```

메타데이터에서 이 두 파일을 불러오도록 다음과 같이 기록한다.

```
latex input: mmd-oblivoir-header
latex input: mypreamble
latex footer: myfooter
latex input: mmd-oblivoir-begin-doc
```

변환된 `.tex` 파일(`foo.tex`)에 대해서 다음 순서로 컴파일한다.

```
$ xelatex foo
$ texindy -L korean -I omega foo.idx
$ xelatex foo
```

앞서 정의한 `\WordIndex` 명령에 옵션을 주려 한다면 MMD 파일에서는 예컨대 다음과 같이 코딩하고

```
=?상위단어?단어=
```

이것을 TEXshop에서 일괄변환할 때

- 찾을 문자열: `\=\?(.*?)\?(.*?)\=`
- 바꿀 문자열: `\\WordIndex\[\1\]\{\2\}`

이런 식으로 하면 될 것이다. 옵션이 뒤에 오는 경우도 비슷하게 할 수 있다.

참고로 스크리브너를 사용한다면 .tex으로 변환하는 과정에서 이 프로그램이 제공하는 강력한 Replacement 기능을 이용할 수 있어 편리하다. 이에 대해서는 내가 쓴 "스크리브너와 라텍" 문서 [1]을 참조하라.

9 사소한 문제 두 가지

9.1 pdflatex에서 에러가 발생하지 않게 하는 법

pdfLaTeX을 조판 엔진으로 사용할 때, 한글은 kotex-utf에 의해 처리된다. kotex-utf 패키지가 가진 성질 중의 하나로 \label에 한글 문자를 쓸 수 없다는 것이 있다. 그래서,

```
# 테스트 #
```

이 코딩은 다음과 같이 변경되는데

```
\section{테스트}
\label{테스트}
```

여기 \label{테스트}에 한글이 사용된 것이 문제를 일으킨다. \unih@ngulchar와 관련된 에러 메시지가 보이는 것은 이 때문이다.

이 문제를 피해가려면 두 가지 방법이 있다.

1. 장절 표제나 label이 달리는 부분에 일절 한글을 쓰지 않는 것이다.
2. 한글이 아닌 ASCII 문자와 숫자로 label을 별도로 달아주는 것이다.

첫 번째 것은 비현실적이다. 두 번째 방법은 예컨대 다음과 같이 한다.

```
# 테스트 [sec:test] #
```

이것은 다음처럼 바뀌기 때문에

```
\section{테스트}
\label{sec:test}
```

에러 없이 컴파일 가능하다. 물론 이 앵커에 대한 조작과 관리 및 참조하기는 문서 작성자 자신의 책임이다.

다행히 XeLaTeX이나 LuaLaTeX은 이러한 불편이 없다. 한글 문서를 XeLaTeX으로 처리하는 것을 원칙으로 하면 이런 불편함이 사라진다.

9.2 한글 이탤릭체

한글에는 이탤릭체가 없다. 그래서 *한글문자*로 코딩해도 (적어도 mmd-oblivoir에서는) 아무런 변화도 나타나지 않는다. 물론 영문자는 이탤릭체로 잘 기울어진다.

한글도 *한글문자*처럼 코딩했을 때 기울어지게 하고 싶다면 mmd-oblivoir-header.tex 헤더 파일을 수정해야 한다. 이 파일을 열어 보면 맨 처음에 다음과 같이 되어 있을 것이다.

```
\documentclass[twoside,openright,
        nanum,amsmath,adjustmath]{oblivoir}
```

여기에 itemph 옵션을 추가하여 저장하라.

```
\documentclass[twoside,openright,
        nanum,amsmath,adjustmath,itemph]{oblivoir}
```

그러나 일반적으로 한글을 기울이는 것은 권장하지 않는다. 차라리 mypreamble.tex 에 다음과 같은 설정을 추가하여 방점을 찍는 편이 나을 것이다.

```
\let\ORIGemph=\emph
\protected\def\emph#1{\dotemph{\itshape#1}}
```

이렇게 하면 이 문서에서와 같이 한글은 방점으로, 그리고 영문자는 *italic*체로 나타나게 할 수 있다.

10 결어

멀티마크다운으로 라텍 문서를 작성하는 방법에 대하여 길게 써보았다. 결국 멀티마크다운이든 라텍이든 글쓰기 도구에 불과하다. 자신의 작업에 가장 잘 집중할 수 있게 해주는 것을 채택하는 것이 현명한 일이다.

다만, 멀티마크다운을 이용하기로 하였다면 이것으로 "모든 것을 하려" 해서는 안 된다. 말 그대로 최종 출력물을 tweaking하는 것은 필수적이다. 그러나 "글쓰기 자체에 집중"하면서 나중에 더 쉽게 출력물을 조절할 수 있게 해주는 도구로서 MMD를 고려해 보는 것은 어떻겠는가.

이 글과 더불어 [1]과 [2]를 더 읽어볼 것을 권한다.

참고 문헌

[1] Nova De Hi (2014),《스크리브너와 라텍》, http://doeun.blogspot.kr/2014/02/blog-post.html.

[2] Nova De Hi (2014),《멀티마크다운 문법》. http://doeun.blogspot.kr/2014/12/blog-post.html. MultiMarkdown 사용자 설명서를 발췌 번역한 것이다.

[3] N. Talbot. "Creating a LaTeX Minimal Example", Dickimaw.

[4] 이기황 (2014), "BibLaTeX을 이용한 한국어 참고문헌 처리의 가능성," 공주대학교 문서작성 워크숍 2014 발표자료, http://wiki.ktug.org/wiki/wiki.php/LaTeXWorkshop/2014?action=download&value=using_biblatex.pdf

찾아보기