

워드 프로세서 사용자를 위한 L^AT_EX

Version 1.2 한국어판

Guido Gonzato, PhD
guido.gonzato at gmail.com

2023-08-28

KTUG 문서화 프로젝트 팀 옮김

한국어판 2023-11-20 r4

한국어판 2025-10-11 r5

요 약

워드 프로세서에 비하여 L^AT_EX으로 문서를 작성하는 것은 여러 가지 이점이 있다. 그러나 처음사용자는 간단한 것이라도 어떻게 해야 하는 것인지, 원하는 기능을 어디서 찾아야 하는지 알기 힘든 경우가 많다. 이 안내서는 워드 프로세서와 L^AT_EX의 문서 작성 방식을 비교함으로써 워드 프로세서를 사용하던 사람들이 L^AT_EX으로 좀 편하게 옮겨갈 수 있게 하려는 시도이다. 주요 워드 프로세서 기능을 열거하고 거기에 해당하는 L^AT_EX 명령을 보인다. 많은 예제를 포함하였다.

한국어판에 관하여

영어판 본문을 우리말로 옮겼다. 예제는 되도록 영문 예제를 그대로 유지하되 역자주를 붙였다. 이 문서는 주로 영어로 된 문서를 pdf_latex으로 컴파일하는 상황을 전제로 작성되어 있으나 오늘날 한국어/한글 문서는 x_el_atex 또는 l_ua_la_tex을 쓰는 것을 권장하고 있으므로 해당 사항에 대해서 역주으로써 적극적으로 언급하였다. 한편 한국어/한글 문서를 ob_livoir로 작성하는 경우에 본문이 소개하는 내용과 다르거나 불필요한 것도 역자주에서 언급하였다. 역자주를 위해서 문서의 레이아웃을 조금 변경하였고 예제 처리 방식을 바꾸었다. 번역문 전체에 대하여 Progress님께서 교정을 보아주셨다.

차 례

		2.4 File>Import(들여오기)	6
		2.5 File>Page Setup(페이지 설정)	9
		2.5.1 Page Setup>Headers and Footers(상하단 면주)	9
		2.6 File>Print Preview(출력 미리보기)	10
		2.7 File>Print(인쇄)	10
		2.8 File>Versions(버전 관리)	10
1	도입	3 Edit 메뉴	10
1.1	사전 지식	3.1 Edit>Autotext(자동완성)	11
1.1.1	예제 보이기에 관하여	4 Insert 메뉴	11
1.1.2	에디터가 지원하는 기능	4.1 Insert>Breaks(행자름)	11
1.1.3	패키지 추가하기	4.2 Insert>Enumerated List(번호 부 나 열문단)	11
1.1.4	Info 페이지 추가	4.3 Insert>Special Character(특수문자)	14
1.2	문서 작성의 황금률		
2	File 메뉴		
2.1	File>New(새 파일)		
2.2	File>Save As...(다른 형식으로 저장)		
2.3	File>Save As Template(템플릿으로 저장)		

4.3.1 유로화 기호(€)	15
4.4 Insert▷Formula(수학식)	15
4.5 SymPy 사용법	16
4.6 Insert▷Footnote(각주)	16
4.6.1 Footnotes and Endnotes(각 주와 미주)	17
4.7 Insert▷Indices(목차)	18
4.8 Insert▷Vertical and Horizontal Space(수평 수직 간격)	18
4.9 Insert▷Tabs(탭)	19
4.10 Insert▷Cross Reference(교차참조)	20
4.11 Insert▷Margin Notes(여백문단)	20
4.12 Insert▷Text Frame(프레임)	20
4.13 Insert▷Image(이미지)	21
4.14 Insert▷Figure(삽도)	22
4.14.1 문단을 파고드는 그림	23
4.15 Insert▷Shapes(도형)	24
4.16 Insert▷Page(페이지)	26
4.17 Insert▷Rule(패션)	26
4.18 Insert▷Hyperlink(하이퍼링크)	26
4.19 Insert▷Comment(주석)	27
5 Format 메뉴	28
5.1 Format▷Line Spacing(행 간격)	28
5.2 Format▷Character(글자 모양)	28
5.2.1 화학반응식의 첨자	29
5.2.2 밑줄 긋기	30
5.2.3 Format▷Character Size(큰 글자)	31
5.2.4 Format▷Character Font(폰트)	31
5.2.5 Format▷Character Colour(글자 색상)	32
5.2.6 Format▷Character Outline(윤곽선 글자)	33
5.3 Format▷Paragraph(문단 모양)	33
5.3.1 Paragraph▷Horizontal Alignment(정렬)	35
5.3.2 Paragraph▷Vertical Alignment(문단 사이의 정렬)	35
5.3.3 Paragraph▷Margins(문단 폭)	35
5.3.4 Paragraph▷Indentation (들여쓰기)	36

5.3.5 Paragraph▷Border and Shade(프레임과 음영)	36
5.3.6 Paragraph▷Colour(문단 색상)	37
5.3.7 Format▷Columns(다단)	37
6 Table 메뉴	38
6.1 Table▷Line Spacing(표의 행 간격)	41
6.2 Table▷Aligning Numbers(수치 정렬)	41
6.3 diagbox(셀의 대각선)	42
6.4 L ^A T _E X 표 생성기	43
6.5 L ^A T _E X 테이블로 데이터 가져오기	43
7 Tools 메뉴	44
7.1 Tools▷Mail Merges(메일 머지)	44
7.2 Tools▷Labels(라벨 작성)	44
7.3 Tools▷Default Language(다국어)	46
7.4 Tools▷Hyphenation(하이프네이션)	46
7.5 Tools▷Spell Check(철자 검사)	47
8 Help 메뉴	47
9 마치는 말	47
A 문서 템플릿	48
B 한글 Markdown과 Pandoc 예제	50

그림 차례

1 이 안내서 저자를 나타내는 스마일리	22
2 Gnuplot 그래프	22
3 두 개의 subfigure를 가진 figure	23
4 L ^A T _E X 수식 입력	25
5 L ^A T _E X이 생성한 개체의 편집	25
A.1 Book template	48
A.2 Report template	48
A.3 Letter template	49
A.4 How to write a notice	49
A.5 How to write a poster	50

표 차례

1 Emacs, Vim, Jed의 단축키 예시	3
2 특수문자를 얻는 방법	14
3 <tabbing> 환경에서 사용되는 명령	19
4 글자 모양	29
5 폰트 크기	30

6	포인트 단위로 나타낸 실제 폰트 크기 . . .	30	8	표준 \LaTeX 환경	34
7	자주 쓰이는 폰트 패밀리 (pdf \LaTeX) . .	33			

1 도입

먼저, 이 소책자는 \LaTeX 입문서가 *아니라*는 점을 말해둔다. 이 문서를 읽고 있다면 최소한 \LaTeX 이 무엇인지 이해하고 있으며 그 기본 명령을 알고 있다고 가정하겠다. 우리가 설명하려고 하는 것은 \LaTeX 을 효과적으로 워드 프로세서 대신 사용하는 방법이다.

워드 프로세서는 친숙한 WYSIWYG 인터페이스 때문에 \LaTeX 보다 쉬운 것으로 보인다. 평균적인 사무원이 사용법을 익히는 데 상대적으로 짧은 시간이면 된다. 문제는 이 물건이 날이 갈수록 느려지고 무거워지고 사용성이 떨어지게 된다는 것이다. 많은 표와 그림을 포함하는 긴 텍스트를 써본 사람은 무슨 말인지 알 것이다.

\LaTeX 은 탁월한 대안이다(때로 그것은 *유일한* 대안이다). 그러나 WYSIWYG에 익숙한 사람들에게 직관적으로 다가오지 않는다.

그러니까 이따금 워드 프로세서 비슷한 기능을 \LaTeX 에서도 써보고 싶을 때가 있다는 것이다. 워드 프로세서 사용법을 알고 있을 때 그와 같은 효과를 \LaTeX 에서도 얻을 수 있다면 좋을 것이다.

그래서 이 안내서를 작성하였다. 이것은 메뉴의 항목과 항목을 대응시키는 설명서이다. 이미 말했듯이 기본적인 \LaTeX 에 대한 지식을 전제로 한다. 만약 그렇지 않고 완전한 초심자라면 다음 문서 중에서 하나를 먼저 읽어볼 것을 권한다.

- [The \(Not So\) Short Introduction to \$\text{\LaTeX}2\epsilon\$ ^{\[1\]}](#)
- [Learn LaTeX in 30 minutes](#)
- <https://en.wikibooks.org/wiki/LaTeX/>

[역주*1] 훌륭한 한국어 번역이 존재한다. 『 $\text{\LaTeX}2\epsilon$ 입문』. TeX Live와 MiKTeX에 포함되어 있으므로 `texdoc lshort-ko` 명령을 내리면 바로 읽을 수 있다.

이 안내서에서는 어떤 가상의 워드 프로세서를 상정하고 그 메뉴와 메뉴 항목을 차례로 살펴보면서 이에 대응하여 같은 일을 하는 \LaTeX 의 방식을 찾아볼 것이다.

1.1 사전 지식

워드 프로세서 기능의 상당 부분은 에디터로 해낼 수 있다. 그리고 일부는 \LaTeX 명령(commands)으로 구현하고 어떤 것은 *패키지*를 사용해야 한다. 패키지라는 것은 \LaTeX 을 확장하여 새로운 명령과 환경을 제공하는 매크로 모음이다. 수많은 패키지들이 있다. 유일한 문제는 그게 어디 있는지, 무슨 일을 하는지, 그리고 어떻게 설치할 수 있는지 알아야 한다는 것이다. 패키지에 대해서는 1.1.3절에서 보충설명한다.

패키지와 그밖의 \TeX 관련 자료들은 Comprehensive TeX Archive Network (CTAN)이라는 네트워크에서 얻을 수 있다. 웹 주소는 <https://www.ctan.org>이다.

이후 CTAN:이라고 표기한 것은 선호하는 CTAN 미러의 \TeX 디렉터리라는 의미로 사용하겠다. 예컨대 CTAN://latex4wp라고 표현한 것은 <https://ctan.org/pkg/latex4wp>에 해당한다.

문서 작성을 위해서는 UTF-8 인코딩을 지원하는 텍스트 에디터가 필요하다. 처음사용자에게는 \LaTeX 통합작업환경(IDE)이 더 편할 수도 있는데 이것은 \LaTeX 소스 작성에 특화된 에디터를 가리키는 말로서 출력 미리보기와 같은 편의도구를 장착하고 있다.

아래 소개하는 에디터 중에 하나를 권장한다. 모두 자유/오픈소스 소프트웨어이다.

- TeXstudio (멀티플랫폼; GNU/Linux AppImage와 Windows용 portable app 이용 가능): <https://www.texstudio.org>

- TeXworks (멀티플랫폼; GNU/Linux AppImage 있음):
<https://tug.org/texworks/> (TeX Live에 포함)
- Texmaker (멀티플랫폼): <https://www.xmlmath.net/texmaker/index.html>
- TeXShop (Mac OS X): <https://www.uoregon.edu/koch/texshop/>
- TeXnicCenter (Windows):
<https://www.texniccenter.org/>

그리고 현대적 \LaTeX 활용을 상정한다. 출력물은 pdf \LaTeX 또는 x \LaTeX , l \LaTeX 으로 생성하는 PDF이다. 마찬가지로 그래픽 포함하기도 더이상 POSTSCRIPT나 EPS를 고려하지 않는다.

1.1.1 예제 보이기에 관하여

이 문서 전체를 통하여 CTAN://[latexdemo](https://ctan.org/latexdemo) 패키지가 제공하는 바 \LaTeX 코드 조각과 그 출력을 보이는 기능을 활용하였다. 이따금 latexdemo가 잘 동작하지 않을 때 예전 CTAN://[example](#) 패키지를 저자가 조금 수정하여 활용한 곳이 있다.^{*2}

[역주*2] 한국어 번역본에서는 원문의 예제 보이기를 따르지 않고 (tcolorbox를 이용하는) 별도의 환경을 작성하여 활용하였다.

1.1.2 에디터가 지원하는 기능

\LaTeX 은 조판기일 뿐이다. 소스의 잘라내기 붙이기, 찾기 바꾸기 등은 에디터가 해주어야 한다. 표 1은 컴퓨터 전문가들 사이에서 유명한 에디터 GNU emacs, vim과 jed의 주요 명령(단축키)을 요약한 것이다. jed는 볼랜드 IDE 키 바인딩으로 설정하였다.

1.1.3 패키지 추가하기

아래 설명은 대부분의 GNU/Linux에 포함되어 있는 \TeX 시스템인 TeXLive에 적용된다. MacTeX에서도 유효할 것이지만 필자는 Linux에 더 익숙하다. 가장 유명한 Windows 배포판인 MiKTeX을 위한 지침은 따로 적어두겠다.

매우 많은 \LaTeX 패키지가 기본적으로 제공된다. 예를 들면 데비안 기반의 GNU/Linux 배포판들은 texlive-*라는 이름의 많은 패키지를 갖추고 있다(여기서 패키지라고 한 것은 .deb 파일 형식으로 제공되는 것을 말한 것으로 \LaTeX 패키지와 혼동하지 않도록 한다).

기본 제공되지 않는 패키지를 설치할 필요가 있다면 다음 절차를 따르라.

1. 다음과 같은 디렉터리 구조를 만든다:

```
$ mkdir -p ~/texmf/tex/latex
```

이 디렉터리 아래 새 패키지를 설치한다.

2. 가까운 CTAN 미러로부터 패키지를 다운로드한다. 대체로 zip으로 압축된 디렉터리 일 것이다. 이것을 foo.zip이라 부르기로 하자.
3. 적절한 위치에 압축을 푼다.

```
$ mkdir ~/texmf/tex/latex/foo
$ mv foo.zip ~/texmf/tex/latex/foo
$ cd ~/texmf/tex/latex/foo ; unzip foo.zip
```

액션	Emacs	Vim	Jed
명령 모드	Alt-X	ESC	Alt-X
삽입 모드	n/a	i a o O	n/a
줄 편집 모드	n/a	:	n/a
파일 조작			
파일 열기	Ctrl-X Ctrl-F	:e	Ctrl-KE
파일 삽입	Ctrl-Xi	:r	Ctrl-KR
파일 저장	Ctrl-X Ctrl-S	:w	Ctrl-KD
새이름으로	Ctrl-X Ctrl-W name	:w name	Ctrl-KS
파일 닫기	Ctrl-XK	:q	Ctrl-KQ
버퍼 변경	Ctrl-XB	bN	Ctrl-KN
취소(undo)	Ctrl-XU	u	Ctrl-U
취소복구(redo)	Ctrl-^	Ctrl-R	Ctrl-G Ctrl-U
끝내기	Ctrl-X Ctrl-C	:qa!	Ctrl-KX
커서 이동			
한 단어 왼쪽	Alt-B	b	Ctrl-A
한 단어 오른쪽	Alt-F	w	Ctrl-F
행처음으로	Ctrl-A	0	Ctrl-QS
행끝으로	Ctrl-E	\$	Ctrl-QD
한 페이지 앞으로	Alt-V	Ctrl-U	Ctrl-R
한 페이지 뒤로	Ctrl-V	Ctrl-D	Ctrl-C
버퍼의 맨처음	Alt-<	1G	Ctrl-QR
버퍼의 맨끝	Alt->	G	Ctrl-QC
지정한 행으로	Alt-G n.	n.G	Ctrl-QI
삭제하기			
왼쪽 한 글자	Ctrl-H	X	BS
오른쪽 한 글자	Ctrl-D	x	Alt-G
왼쪽 한 단어	Alt-DEL	db	Alt-BS
오른쪽 한 단어	Alt-D	dw	Ctrl-T
줄끝까지	Ctrl-K	d\$	Ctrl-QY
현재 행	Ctrl-A Ctrl-K	dd	Ctrl-Y
찾기 & 바꾸기			
찾기	Ctrl-S text	/text	Ctrl-QS
찾아바꾸기	Alt-%	:s/old/new/g	Ctrl-QA
선택영역			
선택 시작	Ctrl-SPACE	v	Ctrl-KB
잘라내기	Ctrl-W	D	Ctrl-KY
복사하기	Alt-W	Y	Ctrl-KH
붙여넣기	Ctrl-Y	P	Ctrl-KC

표 1: Emacs, Vim, Jed의 단축키 예시

- 만약 .sty 파일이 존재하지 않는다면 `latex foo.ins`와 `latex foo.dtx` 명령을 실행하여 그것을 풀어내어야 한다.
- 다음 명령을 실행한다.^{*3}

```
$ texhash ~/texmf
```

[역주*3] 현재의 UNIX TeX Live 시스템에서 `~/texmf`는 `TEXMFHOME`이라는 변수에 할당되어 있다. 따라서 별도로 `texhash`나 `mktexlsr`를 실행하지 않아도 된다. 시스템 관리자로서 `TEXMFLOCAL` 위치에 로컬 패키지를 설치했을 때에는 이 명령의 실행이 필요하다.

MiKTeX에서 패키지를 추가하려면^{*4} MiKTeX console이나 명령행을 이용한다.

`\latex\newpackage` 디렉터리를 `C:\localtexmf\tex\` 아래에 만들고 필요한 파일을 여기에 가져다둔다. 앞에 설명한 절차를 따른 다음 MiKTeX Options를 실행하여 'Refresh now' 버튼을 누른다. 또는 `initexmf -u` 명령을 명령창에서 실행한다. 이것이 전부다.

[역주*4] 이 문단에서 설명하는 MiKTeX 설정 방법은 조금 이전 버전을 위한 것이다. 2023년 현재 MiKTeX Console이라는 응용 프로그램으로 비슷한 일을 할 수 있으나 `C:\localtexmf`는 기본적으로 포함되어 있지 않고 사용자가 직접 추가하여야 하는 등의 변경이 있다.

일단 패키지가 설치된 후에는 `\documentclass` 선언 이후에 다음 행을 덧붙여서 자신의 문서에 활용할 수 있다.

```
\usepackage{foo}
```

1.1.4 Info 페이지 추가

'man'과 'info' 페이지는 소프트웨어의 설명 문서를 명령행 인터페이스로 제공하는 것으로 UNIX나 GNU/Linux 시스템에서 흔히 사용되는 것이다. 만약 `latex2e.info` 문서가 설치되어 있지 않으면 다음과 같이 하라.

- 파일을 다운로드한다.
<https://tug.ctan.org/info/latex2e-help-texinfo/latex2e.info>;
- 다음 명령을 실행한다.

```
$ gzip latex2e.info
$ sudo cp latex2e.info.gz /usr/share/info/
$ sudo ginstall-info latex2e.info dir
```

이제 `info latex2e` 명령으로 `info` 문서를 읽을 수 있다.

1.2 문서 작성의 황금률

무엇보다 다음 사항을 유념해야 한다.

- 문서의 구조화에 익숙해져야 한다. 구조화라는 것은 문서를 편(part)·장(chapter)·절(section)의 체계로 작성하는 것을 말한다. 설령 학술적인 글을 쓰는 때가 아니라도 이것은 유효하다.
- LaTeX은 최고 품질의 출력을 얻을 수 있도록 설계되어 있다. 모양을 결정하는 요소에 신경을 덜 쓸수록 더 좋은 결과를 얻는다.
- 그런데, 이 규칙을 어기고 싶어지는 때가 있을 것이다. 자신이 어떤 규칙을 왜 어기고 있는지 알고 있다면 그래도 된다. 이 안내서에는 이 규칙을 어기는 것에 대한 내용이 제법 많다.

이 간단한 규칙을 적용하면 최종 출력된 결과는 믿을 수 없을 정도로 전문가의 솜씨처럼 보일 것이다.

2 File 메뉴

이 메뉴에 속하는 항목 일부는 \LaTeX 과 무관하다. 예를 들면 File>Open(파일 열기), File>Save(저장하기), File>Close(닫기)들은 에디터의 기능이다.

2.1 File>New(새 파일)

대부분의 워드 프로세서에서 이 메뉴를 선택하면 빈 페이지 하나를 열어준다. \LaTeX 으로 빈 페이지 하나를 만드는 코드는 다음과 같다.

```
\documentclass{article}
\thispagestyle{empty} % 페이지 번호 제거
\begin{document}
% 이 부분은 주석(comment). 뭔가 써둔다.
\end{document}
```

\LaTeX 으로 작성한 문서는 본질적으로 구조적이다. 조금 더 실용적인 예제를 보자.

```
\documentclass[a4paper,12pt]{article}
\begin{document}
\title{My Document}
\author{John Smith}
\date{London, \today}
\maketitle
\begin{abstract}
This is a very short article.
\end{abstract}
\tableofcontents
\listoftables
\listoffigures
\section{First Section}
\label{sec:start}
This is the text of the section. See \cite{Gonzato} for details.
\section{End}
\label{sec:end}
This is the end of the document. Please go to Section
\ref{sec:start} to read it again.
\begin{thebibliography}{99}
\bibitem{Gonzato} Gonzato G. \textit{\LaTeX} for Word Processor
Users}. CTAN, 2001--2023.
\end{thebibliography}
\end{document}
```

부록의 A절에 더 많은 문서 템플릿이 마련되어 있다.

2.2 File>Save As...(다른 형식으로 저장)

\LaTeX 문서를 다른 포맷으로 변환하고자 할 때 사용할 수 있는 것들이다.

- \TeX 4ht는 \LaTeX 을 HTML/XML로 변환하기 위한 최선의 프로그램일 것이다.^{*5}
<https://tug.org/tex4ht>

[역주*5] \TeX Live에 포함된 `lwarp`라는 패키지도 살펴볼 수 있다.

- HTML로 변환하기 위해 latex2html을 이용할 수도 있다.

[CTAN://latex2html](http://ctan.org/latex2html)

- latex2rtf는 RTF (Rich Text Format)로 변환한다.

[CTAN://latex2rtf](http://ctan.org/latex2rtf)

- detex (명령행 변환기)는 \LaTeX 태그를 제거하여 플레인 텍스트로 변환해준다.

<https://github.com/pkubowicz/opendetex>,

[CTAN://detex/](http://ctan.org/detex/)

그리고 Pandoc이라는 프로그램이 있다. 이 프로그램에 대해서는 별도의 절을 할애하여 설명하겠다. 2.4절을 읽어보기 바란다. PDF 제작과 미리보기에 대한 사항은 2.6절을 보라.

2.3 File▷Save As Template(템플릿으로 저장)

\LaTeX ‘템플릿’이라는 것이 새로운 \LaTeX 패키지를 만드는 것을 의미한다면, 이 안내서가 다루는 범위를 넘어서는 복잡한 문제이다.

2.4 File▷Import(들여오기)

Pandoc 홈페이지 <https://pandoc.org>에 다음과 같은 내용이 있다.

마크업 포맷의 파일을 다른 포맷으로 변환하려 한다면 pandoc이 만능 도구^{*6}가 된다.

[역주*6] “맥가이버 칼”(swiss army knife)

Pandoc은 \LaTeX 문서를 생성하는 훌륭한 변환기이며, 그것에만 그치지 않는다. 다양한 포맷의 문서를 작성하거나 변환하는 데 필수적인 프로그램이다.

Pandoc의 기본 포맷은 Markdown 확장이다. 이 포맷으로부터 \LaTeX 으로의 변환은 쉽게 이루어진다. 예를 들어보자. 다음에 보이는 것은 Markdown으로 작성한 문서이다.

```
---
title: |
  This is the title: \
  now write the rest
author: Guido Gonzato
date: August 2023
abstract: |
  This is the abstract.
...

<!-- This is a comment. -->

[comment]: # (This too is a comment)

# Section

This is bold text, this is emphasized text, this is normal text.

If you're bored, go to [The End].

## Subsection

This is verbatim text.
```

```
# The End
```

```
My dad used to say:
```

```
> Damn, Pandoc has not been invented yet!
```

```
but not it has.
```

```
<!-- end of pandoc_template.md -->
```

이 문서를 컴파일 가능한 \LaTeX 소스로 바꾸려면 다음 명령을 내린다.

```
$ pandoc -s pandoc_template.md -o pandoc_template.tex
```

\LaTeX 을 거쳐 이 소스를 직접 PDF로 인쇄하려면 다음 명령을 내린다.

```
$ pandoc --toc \  
-V urlcolor=blue -V toccolor=red \  
-V geometry:margin=2cm \  
pandoc_template.md -o pandoc_template.pdf
```

-V 태그로 시작하는 행은 옵션을 지시하는 것이다. 상세한 것은 Pandoc 설명서를 보라.

Pandoc을 이용하면 다양한 포맷의 문서를 \LaTeX 으로 변환할 수 있다. 몇 가지 예를 들어보겠다.

```
$ pandoc -s index.html -o index.tex  
$ pandoc -s text.docx -o text.tex  
$ pandoc -s text2.rtf -o text2.tex  
...and so on.
```

-s 플래그는 “standalone”의 의미이다. 즉 출력 파일은 완전한 \LaTeX 문서가 된다는 뜻이다.

Markdown에서 수식과 한글

이 소절은 역자가 추가한다. Markdown에는 여러 변종이 있고 각자 제공하는 기능이 조금씩 다르다. 수학적 지원에 각 Markdown 사양마다 약간 차이가 있는데, pandoc Markdown을 기준으로 수학적식은 다음과 같이 처리한다.

- 행중(in-line) 수식은 $_$ 로 시작하고 $_$ 로 끝난다. 즉 행중 수식의 끝은 $\$$ 기호와 스페이스이다.
- 별행(displayed) 수식은 $\$$ 행과 $\$$ 행 사이에 적어넣는다.

행중 수식의 끝이 $_$ 와 같이 스페이스로 끝나야 한다는 점 때문에 한국어 문서 작성에서 약간의 곤란을 겪을 수 있는데, 한국어의 조사는 앞말에 이어쓰기 때문이다. 부득이하게 조사를 행중 수식과 떨어뜨려놓아야 한다는 문제가 있다.⁷

여러 줄 수식을 써야 한다면 amsmath의 $\langle aligned \rangle$ 와 같이 $\langle equation \rangle$ 내부에서 쓸 수 있는 여러 줄 수식 환경을 이용할 수 있다.

한글 Markdown을 \LaTeX 을 거쳐 PDF로 변환하는 데 관련된 샘플을 부록 B에 실었다.

[역주⁷] 참고로 Markdown의 변종 가운데 하나인 Multimarkdown은 $\backslash($ 와 $\backslash)$ 라는 행중 수식 표시방법을 지원하기 때문에 조사를 $\backslash)$ 뒤에 잇대어 쓰는 것이 가능하다.

This is the title:
now write the rest

Guido Gonzato

January 2018

Abstract

This is the abstract.

Contents

Section	1
Subsection	1
The End	1

Section

This is **bold text**, this is *emphasized text*, this is normal text. If you're bored,
go to **The End**.

Subsection

This is `verbatim text`.

The End

My dad used to say:

 Damn, Pandoc has not been invented yet!
but not it has.

2.5 File▷Page Setup(페이지 설정)

용지 크기, 인쇄 방향, 여백을 설정하는 일반적 방법은 \documentclass에 옵션 파라미터를 주는 것이다. 용지 크기는 a4paper, a5paper, b5paper, letterpaper, legalpaper, executivepaper 중의 하나를 지정한다. 방향은 portrait가 기본인데, landscape를 부여하여 가로가 긴 용지가 되도록 바꿀 수 있다. 예를 들면 다음과 같다.

```
\documentclass[a5paper,landscape,12pt]{article}
```

문서 전체에 걸친 마진은 \setlength 명령으로 설정한다. 이 명령은 변수와 카운터의 값을 바꿀 때 사용하는 것이다.^{*8} geometry 패키지를 이용하는 것이 더 좋다. 이 패키지를 이용하면 용지 사이즈, 마진 길이와 같은 파라미터들을 완전히 제어할 수 있다. geometry에는 너무 많은 선택사항이 있어서 여기에 다 열거할 수 없다. 패키지 문서를 읽어보는 것을 권한다. 아래 예에서 열추 사용법을 다 보였는데 일부 파라미터는 다른 것과 동시에 사용할 수 없는 것이 있으며, 단지 예를 들기 위해 보인 것도 있다.^{*9}

```
\usepackage{geometry} % top of document
...
\geometry{paperwidth=25cm}
\geometry{paperheight=35cm}
% or: \geometry{papersize={25cm,35cm}}
\geometry{width=20cm} % total width
\geometry{height=30cm} % total height
% or: \geometry{total={20cm,30cm}}
\geometry{textwidth=18cm} % width - marginpar
\geometry{textheight=25cm} % height - header - footer
% or: \geometry{body={18cm,25cm}}
\geometry{left=3cm} % left margin
\geometry{right=1.5cm} % right margin
% or: \geometry{hmargin={3cm,1.5cm}}
\geometry{top=2cm} % top margin
\geometry{bottom=3cm} % bottom margin
% or: \geometry{vmargin={2cm,3cm}}
\geometry{marginparwidth=2cm}
\geometry{head=1cm} % header space
```

\geometry 명령 대신 다음처럼 패키지 옵션을 주는 방법도 있다.

```
\usepackage[left=3cm, right=2cm]{geometry}
```

[역주*8] 이 문서의 저자는 counter와 dimension을 엄격히 구분하지 않고, 파라미터 변수를 일괄해서 “카운터”라고 지칭하고 있다. 이 대목에서는 원문을 그대로 옮기되, 이후로 번역본은 정수 카운터에 대해서만 카운터라고 부르고 다른 변수는 맥락에 따라 번역하였다.

[역주*9] memoir/oblivoir에서는 geometry와는 다른 방식의 클래스 자체 페이지 레이아웃 설정 방법이 있다. oblivoir가 제공하는 fapapersize 패키지도 간편하게 쓸 수 있다. 패키지 문서를 참고하라. geometry가 memoir/oblivoir에서 충돌을 일으키지는 않는 것으로 알려져 있으나 클래스 자체 용지 설정과 동시에 사용할 수는 없다.

2.5.1 Page Setup▷Headers and Footers(상하단 면주)

fancyhdr 패키지를 사용하면 \pagestyle{fancy}라는 명령을 쓸 수 있게 된다.^{*10} 문서의 상단에 현재 절 — book.cls라면 장(chapter) — 을 알려주는 상단면주와 하단에 페이지 번호를 나타내는 하단면주를 제법 예쁜 모양으로 만들어준다. 상하단 면주를 사용자화하는 것도 가능하다. 각 면주 영역을 왼쪽, 가운데, 오른쪽의 세 부분으로 구성하는데, 각 부분에 왼쪽 정렬, 가운데 정렬, 오른쪽 정렬이 각각 적용된다. 다음 보기를 보자.

[역주*10] memoir/oblivoir에서는 fancyhdr와는 다른 방식의 클래스 자체 ‘페이지 스타일’ 설정 방식이 있다. fancyhdr와 충돌을 일으키지는 않는 것으로 알려져 있으나 클래스의 설정 방식과 동시에 사용할 수는 없다.

```

\usepackage{fancyhdr}
...
\lhead{} % empty
\chead{Hello, world!}
\rhead{Page \thepage} % page number
\lfoot{}
\cfoot{\textbf{Hello!}}
\rfoot{}

```

(역자) 이 예제에서 제공하는 명령어 예시는 예전 fancyheading의 것이다. 현재의 fancyhdr도 이 명령을 지원하기는 하지만 되도록 \fancyhead, \fancyfoot 명령으로 설정하는 것이 좋다. 특히 양면 문서를 디자인할 때에 그러하다.

2.6 File>Print Preview(출력 미리보기)

너무나 간단하다. pdf_latex 또는 xelatex, lualatex으로 PDF 파일을 바로 만들어서 자신이 즐겨 쓰는 PDF 미리보기 앱을 실행하면 된다.

hyperref이나 url 같은 패키지를 쓰면 브라우징가능한 PDF 파일을 만든다. 이에 대해서는 4.18절을 보라. 그러나 pdf_latex에서 일부 패키지를 사용할 때 문제를 겪을 수도 있는데, ¹¹ 이와 관련한 자세한 사항은 4.14절에서 다룬다.

[역주¹¹] 저자는 EPS 그림 삽입을 염두에 둔 것으로 보인다. 관련 사항을 언급하고 있지 않으므로 역자가 부연하면, EPS 그림을 원칙적으로 pdf_latex에서 삽입할 수 없으나, 현재의 TeX Live 시스템은 사용자가 의식하지 않아도 EPS 그림을 PDF로 변환하여 포함해주는 과정이 자동화되어 있다.

2.7 File>Print(인쇄)

자신이 사용하고 있는 PDF 미리보기 앱의 메뉴에 File>Print 항목이 있을 것이다.

2.8 File>Versions(버전 관리)

변경 이력 관리와 공동 작업을 위한 프로그램이 몇 가지 있다. 옛날 스타일을 선호하는 컴퓨터 전문가라면 RCS같은 단일 사용자 툴을 좋아할 수도 있지만 그보다 훨씬 강력하고 다중 사용자를 지원하는 Subversion, Git, Mercurial 같은 도구가 더 낫다. Git이 가장 많이 사용되는 것인데 L^AT_EX 통합 관리에 관해서는 다음 문서를 참고하라.

- <https://www.desy.de/~bargheer/gitintro/git.html>
- <https://www.math.cmu.edu/~gautam/sj/blog/20130929-git-quickstart.html>

3 Edit 메뉴

이 메뉴에 해당하는 사항은 L^AT_EX의 기능이 아니라 편집기와 관련된 것이 많다. 먼저 표 1에서, 잘라내기·복사하기·붙이기·찾기·찾아바꾸기(Edit>Cut, Edit>Copy, Edit>Paste, Edit>Find, Edit>Replace) 등에 해당하는 몇 가지 에디터의 키바인딩을 살펴보자.

자르거나 붙여넣거나 하기 위해서이기도 하지만 일부분에 특정 스타일을 적용하려고 텍스트를 선택할 때가 있다. L^AT_EX에서 이에 해당하는 액션은 중괄호로 둘러싸거나 환경 안에 넣는 것이다. 예를 들면 일부 텍스트에 굵은 글자 속성을 부여하려면 다음 예시 중 한 가지 방법으로 한다.

```
this is \textbf{bold text;}\\
this is also
{\bfseries bold text;}\\
\begin{bfseries}
this is bold text, too!
\end{bfseries}
```

this is **bold text**;
this is also **bold text**;
this is bold text, too!

3.1 Edit▷Autotext(자동완성)

여기서 자동완성이라 하는 것은 예컨대 'PS'라고 타이핑하면 'PostScript'라고 자동으로 확장하여 입력해주는 것을 말한다. 에디터가 이런 일을 해준다. \LaTeX 에서 이에 해당하는 것은 대략 다음과 같은 것이다.

```
\newcommand{\PS}{\textsc{PostScript}}
```

이렇게 정의한 다음에 \PS라고 입력하는 것은 \textsc{PostScript}라고 입력하는 것과 같다. 대소문자 구별이 중요하다.

4 Insert 메뉴

4.1 Insert▷Breaks(행자름)

- 행이 나누어지지 않는 공백문자는 ~ (틸데)로 표현한다.
 - 강제 행 나누기는 \linebreak나 \newline을 쓴다. 둘의 차이는 아래 보기를 보라.
또는 \\로 강제 행 나누기를 할 수 있는데 이 때는 행 간격을 옵션으로 줄 수 있다(예: \\[1cm]).^{*12}
 - 새로운 문단을 시작하려면 빈 줄을 두거나 \par 명령을 쓴다.
 - 끝으로 페이지를 나누는 \newpage 또는 \clearpage 명령이 있다.
- \newline과 달리 \linebreak는 현재 행을 줄끝까지 채운다.

[역주*12] 저자는 \\를 \par와 완전히 동일한 것으로 소개하고 있으나 역자는 그렇게 생각하지 않기 때문에 이 항목으로 옮겼다. 한편 \\는 평문단에서는 (옵션을 줄 수 있는) \newline과 거의 같지만 배열(array, tabular, tabbing 등)에서는 현재 행(row)의 종료를 나타낸다.

```
I am stretched!\linebreak
But I am not.\newline
Another line.\\
Ok, now you get it.
```

I am stretched!
But I am not.
Another line.
Ok, now you get it.

한편 \clearpage가 \newpage와 다른 점은 현재 위치까지 보류되어 있는 플롯(그림이나 표)들을 모두 출력한 다음에 페이지를 나눈다는 것이다. 플롯에 대해서는 4.14절에서 설명한다.

4.2 Insert▷Enumerated List(번호부 나열문단)

나열형 문단(list)에는 문단 머리에 점(bullet)을 붙이는 <itemize>와 번호를 붙이는 <enumerate> 환경이 있다.^{*13} \item 명령에 옵션 인자를 부여하여 나열형 문단의 문단 머리 기호를 다른 모양으로 바꿀 수 있다.

[역주*13] <description>은 문단 머리에 (옵션 인자로 주어진) 텍스트를 두고 설명 형식의 문단을 만든다.

<pre> \begin{itemize} \item[*] with an asterisk; \item[-] with a dash; \item[.] with a dot. \end{itemize} </pre>	<pre> * with an asterisk; - with a dash; . with a dot. </pre>
--	---

각 항목에 표시되는 항목 머리(숫자)의 모양을 바꾸는 또다른 방법은 리스트의 1부터 4수준까지 관련 카운터(counter)¹ 스타일을 재정의하는 것이다. `\labelitemi`, `\labelitemii`, `\labelitemiii`, `\labelitemiv`는 `itemize` 환경에 해당하고, `\labelenumi`, `\labelenumii`, `\labelenumiii`, `\labelenumiv`는 `enumerate`의 라벨 카운터 스타일이다.

카운터 스타일에 몇 종류가 있다. `\arabic`은 ‘일반’ 숫자이고 `\roman`은 소문자 로마 숫자(예를 들면 8을 `viii`로 표기), `\Roman`은 대문자 로마 숫자, `\alph`와 `\Alph`는 각각 알파벳 소문자와 알파벳 대문자이다.^{*14} 그밖에 `\fnsymbol`이라는 것도 있는데 이에 대해서는 나중에(17페이지) 설명하겠다.

[역주*14] `koTeX`과 `oblivoir`는 한글식 카운터 수식 명령을 제공한다. 그 중에는 `\gana`, `\pgana`, `\ogana`, `\onum`, `\pnum` 등이 있다. 각각 카운터 값이 1일 때 가, 개, ㉠, ①, (1)로 출력한다.

<pre> \begin{itemize} \renewcommand{\labelitemi}{*} \renewcommand{\labelitemii}{-} \item first level, item 1 \item first level, item 2 \begin{itemize} \item second level, item 1 \item second level, item 2 \end{itemize} \item first level, item 3 \end{itemize} </pre>	<pre> * first level, item 1 * first level, item 2 - second level, item 1 - second level, item 2 * first level, item 3 </pre>
---	--

`enumerate` 리스트에서 로마 숫자와 대문자 알파벳을 쓰려면 다음과 같이 한다.

```

\begin{enumerate}
\renewcommand{\labelenumi}{\Alph{enumi}}
\renewcommand{\labelenumii}{\Roman{enumii}}
  \item first level, item 1
  \item first level, item 2
  \begin{enumerate}
    \item second level, item 1
    \item second level, item 2
  \end{enumerate}
  \item first level, item 3
\end{enumerate}

```

```

A first level, item 1
B first level, item 2
  i second level, item 1

```

¹`TeX`에 의하여 번호가 붙는 장절, 리스트, 그림과 같은 텍스트 요소는 각자 카운터를 가지게 되어 있다.

ii second level, item 2
C first level, item 3

또다른 방법으로 `enumerate` 패키지를 쓰는 방법이 있다.^{*15} `<enumerate>` 환경을 재정 의하여 옵션 인자로 항목 번호 스타일을 지정할 수 있게 하였다. A a l i 1 가운데 한 글자를 옵션 인자로 지정하면 각 카운터의 형식이 각각 `\Alph`, `\alph`, `\Roman`, `\roman`, `\arabic` 으로 나타난다.^{*16} 항목 머리에 번호 이외의 텍스트를 추가하려면 중괄호로 묶어준다.

[역주*15] `enumerate` 패키지보다 더 풍부한 기능을 제공하는 `enumitem` 패키지도 있다.

[역주*16] `memoir` 클래스는 `enumerate` 패키지의 기능을 포함하고 있으므로 별도로 로드할 필요가 없다. `oblivoir`는

```
\begin{enumerate}[{Example} I.]
  \item First example.
  \label{item:first}
  \item Second example.
  \item Last example.
  Go to Item~\ref{item:first}.
\end{enumerate}
```

Example I. First example.
Example II. Second example.
Example III. Last example. Go
to Item I.

가, (가), ㉠, ㉠, ㉠, (1), ①, (a), @

라는 항목 머리 지시자를 추가하고 있다. `oblivoir`가 아니라면 `koTeX` 추가 패키지인 `dhucs-enumerate`나 `dhucs-enumitem`을 참고하라.

항목 머리 숫자 자체를 바꾸려면 카운터를 재정의하라.

```
\begin{enumerate}
  \setcounter{enumi}{2}
  \item Example 3.
  \item Example 4.
  \setcounter{enumi}{5}
  \item Example 6.
\end{enumerate}
```

3. Example 3.
4. Example 4.
6. Example 6.

번호 붙은 리스트를 문단 내부에서 구현하려면 `paralist` 패키지를 사용하면 된다. `<inparaenum>` 환경을 제공한다.

```
I'll throw in a list of items:
\begin{inparaenum}
  \item apples,
  \item pears, and
  \item oranges.
\end{inparaenum}
The same list can be labelled with letters:
\begin{inparaenum}
  [\itshape a) \upshape]
  \item apples, \label{first}
  \item pears, and
  \item oranges. The first item is \ref{first}.
\end{inparaenum}
```

I'll throw in a list of items: 1. apples, 2. pears, and 3. oranges. The same list can be labelled with letters: a) apples, b) pears, and c) oranges. The first item is a.

[역주*17] `oblivoir` 부수 패키지인 `xob-paralist`는 앞서 `enumerate`에 대해서 했던 것과 마찬가지로 한글 문장을 위한 항목 머리 지시자를 추가해준다.

위의 예에서 보듯이 A a l i 1 문자가 옵션 인자로 사용되면 카운터 수식자로 동작한다. `paralist`는 이밖에도 다양한 일을 할 수 있으므로 패키지 문서를 읽어볼 것을 권한다.^{*17}

특수 문자	L ^A T _E X 입력 시퀀스
\$	\\$ or \textdollar
&	\&
%	\%
_	_ or \textunderscore
{	\{ or \textbraceleft
}	\} or \textbraceright
<	\$<\$ or \textless
>	\$>\$ or \textgreater
\	\textbackslash
	\textbar
•	\textbullet
‡	\textdaggerdbl
†	\textdagger
¶	\textparagraph
§	\textsection
©	\textcopyright
^	\textasciicircum
~	\textasciitilde or \~{ }
~	\$\sim\$
®	\textregistered
™	\texttrademark
ª	\textordfeminine
º	\textordmasculine




표 2: 특수문자를 얻는 방법

4.3 Insert▷Special Character(특수문자)

먼저, 몇 가지 문자는 L^AT_EX에서 특별한 의미를 가진다는 것을 기억하자. 이 문자들은 앞에 \를 붙이거나 수학 모드에서 입력하거나 아니면 특정 매크로를 사용하여야 한다. 표 2를 보라.

특수문자를 입력하는 다른 방법으로 ASCII 코드와 \char 명령을 쓰는 것이 있다. 예를 들면 \$&^~과 같은 문자들을 각각 \char36 \char38 \char94 \char126이라고 입력하여 얻을 수 있다.

수많은 특별한 문자나 기호를 제공하는 패키지도 있다. 예를 들면 **pifont** 패키지는 \ding과 \dingfill, \dinglint, \dinglist 명령을 갖추고 있다. \ding 명령은 특정 코드의 덩벙(Dingbat) 문자를 찍는다. \dingfill과 \dingline은 \fill과 \line에 대응한다. \dinglist 환경은 인자로 주어진 덩벙 코드를 항목 머리로 사용하는 리스트를 만든다.

<pre>\begin{dinglist}{43} \item one \item two \item three \end{dinglist}</pre>	 one  two  three
--	---

다음 보기의 dingautolist 환경은 주어진 인자에 해당하는 덩벙 코드부터 시작하여 리스트를 만드는 멋진 기능을 제공한다.

```
\begin{dingautolist}{172}
\item one
\item two
\item three
\end{dingautolist}
```

- ① one
- ② two
- ③ three

기호 문자는 너무 많아서 여기서 다 언급할 수 없다. 그 대신 ‘The Comprehensive L^AT_EX Symbol List’라는 문서를 소개하러 한다. CTAN://comprehensive.^{*18}

[역주*18] TeX Live를 설치했다면 명령행에서 `texdoc symbols` 또는 `texdoc comprehensive` 명령을 내려보라.

4.3.1 유로화 기호(€)

유로화 기호는 `eurosym` 패키지로 식자할 수 있는데, 다음과 같은 두 가지 방식으로 사용 가능하다.

```
\usepackage[gen]{eurosym}
\usepackage[official]{eurosym}
```

두 경우 모두 `\euro` 명령을 제공하고 그 결과는 €와 같다. 이 기호가 인쇄되는 모양이 옵션 선언에 따라 달라진다. `[gen]` 옵션을 주면 폰트 디자인에 따르는 유로화 기호를 얻는다. 반면 두 번째 보기의 선언은 €라는 모양을 인쇄한다. 미묘한 차이가 있다. `\officialeuro` 명령으로 어떤 상황에서든지 이 ‘official euro’ 모양을 얻을 수 있다.^{*19}

[역주*19] 원화와 엔화를 나타내는 기호 ₣, ¥을 `\textwon`, `\textyen`으로 얻을 수 있다.

유로화 기호는 `marvosym` 패키지에서도 제공하는데, 이 패키지는 이밖에 수많은 멋진 기호들을 사용하게 한다. `\EUR` 명령으로 €를 인쇄할 수 있다.

4.4 Insert▷Formula(수학식)

L^AT_EX은 특히 수식의 조판에 강점이 있다. 수학 기호를 텍스트의 일부로 — *인라인 모드* (*inline mode*) — 적으려면 \$ 부호로 그 시작과 끝을 나타낸다.

```
I like math: $x^n + y^n \neq
z^n \ \forall n \neq 2$
is my favourite theorem.
```

I like math: $x^n + y^n \neq z^n \forall n \neq 2$
is my favourite theorem.

`<displaysmath>`와 `<equation>` 환경은 수학식을 본문과 별도 문단으로 조판한다. 이것을 *디스플레이 모드* (*display mode*)라고 부른다.^{*20}

[역주*20] `amsmath` 패키지를 로드하면 `\eqref` 명령을 쓸 수 있다. 다음 예에서 “Eq.~\ref{eq:fermat}”라고 한 부분을 `\eqref`으로 바꾸면, Eq. (1)과 같이 나타난다.

```
Fermat's Last Theorem is
defined as:
\begin{equation}
x^n + y^n \neq
z^n \ \forall n \neq 2
\label{eq:fermat}
\end{equation}
Can you prove
Eq.~\ref{eq:fermat}?
```

Fermat’s Last Theorem is defined
as:

$$x^n + y^n \neq z^n \forall n \neq 2 \quad (1)$$

Can you prove Eq. 1?

4.5 SymPy 사용법

프로그래밍 언어와 스프레드시트는 수식 입력에 표준 문법을 사용한다. SymPy라는 파이썬 라이브러리를 이용하면 수학 표현식으로부터 \LaTeX 수식을 얻을 수 있다. 파이썬에 정통할 필요가 없다. 단지 표준적인 수학식 문법을 쓰면 된다.

다음은 SymPy 예제이다. 우리는 수식 $\left(\frac{x}{2} - \frac{2y}{3}\right)^3$ 과 그것을 전개한 $\frac{x^3}{8} - \frac{x^2y}{2} + \frac{2xy^2}{3} - \frac{8y^3}{27}$ 에 해당하는 \LaTeX 코드를 얻고자 한다.

```
In [1]: from sympy import *

In [2]: init_session()
IPython console for SymPy 1.9 (Python 3.10.6-64-bit) (ground
types: python)

These commands were executed:
>>> from __future__ import division
>>> from sympy import *
>>> x, y, z, t = symbols('x y z t')
>>> k, m, n = symbols('k m n', integer=True)
>>> f, g, h = symbols('f g h', cls=Function)
>>> init_printing()

Documentation can be found at https://www.sympy.org

In [3]: expr = (x/2 - 2*y/3)**3

In [4]: latex(expr)
Out[4]: '\\left(\\frac{x}{2} - \\frac{2 y}{3}\\right)^{3}'

In [5]: latex(expr.expand())
Out[5]: '\\frac{x^{3}}{8} - \\frac{x^{2} y}{2} + \\frac{2 x}{3} y^{2} - \\frac{8 y^{3}}{27}'
```

이제 위의 결과로부터 얻어지는 수식을 에디터로 복사해 붙여넣기한다. 단 $\backslash\backslash$ 를 \backslash 로 바꾸기만 하면 된다.

만약 파이썬 언어에 능통하다면 python \LaTeX 패키지와 LaTeX Expression project 모듈을 살펴보는 것도 좋겠다.

4.6 Insert>Footnote(각주)

필요한 것은 $\backslash\text{footnote}[n]\{\text{footnote text}\}$ 라는 명령뿐이다. 옵션 인자인 $[n]$ 는 각주 번호를 바꾼다. $\backslash\text{footnote}$ 명령은 단어와 마침표, 쉼표, 기타 문장부호 뒤에 두어야 한다.

번호 대신 기호 또는 임의의 텍스트를 각주 기호로 사용하려면 $\backslash\text{footnote}$ 카운터를 재정의한다.

```
\renewcommand{\thefootnote}%
{read me!}
This footnote\footnote

{I mean this one.}
says it all.
```

This footnote^{read me!} says it all.

read me!^I mean this one.

이 방법을 사용하면 각주 번호를 로마숫자나 기호문자로 나타낼 수 있다.

```
\renewcommand{\thefootnote}
{\Roman{footnote}}
This\footnote{The first.}
is the first footnote,
and this\footnote{The second.}
is the second.
\renewcommand{\thefootnote}
{\fnsymbol{footnote}}
The end.\footnote[8]{At last!}
```

This^I is the first footnote, and this^{II} is the second.

The end.*

^IThe first.
^{II}The second.
*At last!

\fnsymbol{footnote}라고 한 부분을 잘 보자. footnote 카운터가 1...9의 범위일 때 각각 다음 아홉 개의 기호문자가 표현된다. * † ‡ § ¶ || ** †† ‡‡

동일한 각주에 대해서 서로 다른 참조 각주 번호를 달려 할 때에는 직접 각주 번호를 써넣지 말고 다음과 같이 하라.

```
This\footnote{the first.}
\newcounter{myfootnote}
\setcounter{myfootnote}
{\value{footnote}}
and that\footnote{the second.}
are footnotes: please read note
\footnotemark
[\value{myfootnote}] again.
```

This¹ and that² are footnotes: please read note¹ again.

¹the first.
²the second.

주의: minipage 내부에서는 본문과는 별도의 각주 카운터를 갖는데, mpfootnote이다. \thempfootnote로 그 값을 인쇄할 수 있다.

4.6.1 Footnotes and Endnotes(각주와 미주)

endnotes 패키지는 \endnote라는 새로운 명령을 제공한다. 이 명령을 \footnote 자리에 대신 쓰면 페이지 하단에 각주를 붙이는 대신 \theendnotes 명령이 놓이는 위치에 모아서 프린트한다.¹ 장이나 절의 마지막에 주석을 모아서 인쇄하려 할 때에 유용하다.

\footnote 명령으로 입력되어 있는 각주를 문서의 맨끝에 가져다 놓을 수도 있다. 그러기 위해서는 프리앰블에 다음 행을 추가해준다.

```
\let\footnote=\endnote
```

그리고 아래와 같은 세 줄의 명령을 문서 마지막에 적어넣는다.^{*21}

[역주*21] endnotes를 식자할 때 붙는 타이틀 매크로는 \notesname이고 기본값이 “Notes”로 되어 있다. 여기서는 한국어로 표현하기 위해 이 매크로를 renewcommand 하였다.

```

\newpage
\begingroup
\parindent 0pt
\parskip 2ex
\def\enotesize{\normalsize}
\renewcommand\notesname{미주} %% 역자 추가
\theendnotes
\endgroup

```

결과는 다음과 같이 나타난다.

미주

¹이것은 미주이다.

4.7 Insert▷Indices(목차)

내용 목차, 표 목차, 그림 목차 등을 생성하는 것은 \LaTeX 에서 너무나 간단한 일이다. 다음과 같은 명령을 문서의 첫 번째 `\section` 또는 `\chapter` 명령이 나오기 전에 적어넣으면 된다.

```

\tableofcontents
\listoffigures
\listoftables

```

4.8 Insert▷Vertical and Horizontal Space(수평 수직 간격)

내가 아는 한 워드 프로세서에는 이러한 기능이 없다. 워드 프로세서의 한계인데 \LaTeX 으로는 아주 간단하게 할 수 있는 일이다.

간격 채우기는 텍스트를 수직으로, 수평으로 또는 수직/수평으로 중앙정렬하는 데 사용할 수 있다. 워드 프로세서에서 이것은 꽤 곤란한 작업이어서 상당한 시행착오를 통해 해야 했다. `\null` 명령이나 `~`로 고정점 위치를 설정하고 `\vfill`과 `\hfill` 명령을 쓰면 수직 수평 채우기가 이루어진다. 다음 보기를 보라.

one \hfill two\\	one	two
\vfill		
~ \hfill three \hfill		
↔ ~\\	three	
\vfill		
four \hfill five	four	five
\null		

일반적으로 \LaTeX 에서는 둘 이상의 공백을 의도적으로 넣어도 그것을 반영하지 않는다. 둘 이상의 공백은 그냥 하나의 공백으로 간주한다. 그러나 문서의 품위가 어찌 되든 상관없고 꼭 둘 이상의 공백을 강제로 넣어야 한다면, “잘라지지 않는 공백”인 `~` 부호를 사용하라.

길이를 지정하여 공백을 넣으려 한다면 `\hspace` 명령을 쓴다.^{*22}

[역주*22] 수직 간격에 대하여 `\vspace`를 같은 방법으로 쓸 수 있다. 다만 `\vspace`는 문단이 종료된 다음에 (테크니컬하게 말하자면 `vertical mode`에서) 써야 한다.

표 3: <tabbing> 환경에서 사용되는 명령

명령	동작
\=	탭 스톱 위치를 설정한다.
\>	다음 탭 스톱 위치로 이동한다.
\+	탭 스톱 위치를 한 칸씩 오른쪽으로 민다.
\-	탭 스톱 위치를 한 칸씩 왼쪽으로 당긴다.
\\	행의 끝
\pushtabs	모든 탭 스톱 위치를 저장한다.
\poptabs	저장했던 탭 스톱 위치를 불러온다.

This is a \hspace{2cm}
2-cm-wide~~~hole.

This is a 2-cm-wide hole.

4.9 Insert▷Tabs(탭)

<tabbing> 환경은 TAB 키를 눌렀을 때의 동작과 유사한 결과를 제공한다. 텍스트를 열(column) 위치로 배치하는 데 사용한다. 표 3은 이 환경에서 주로 쓰이는 명령을 요약하고 있다.

다음 코드는 명령 사용법의 예시이다.

```
\begin{tabbing}
% 탭 위치를 설정
~ \hspace{1cm} \= ~ \hspace{1.7cm} \=
~ \hspace{2.2cm} \= \kill % 이 행은 표시하지 않음
Zero \> One \> Two \> Three \\
Zero \> One \> \> Three \+ \\ % 한 칸씩 오른쪽 이동
Zero \> Two \> Three \- \\ % 한 칸씩 왼쪽 이동
Zero \> One \> Two \\
\pushtabs % 탭 위치 저장
new tab 1{\dots} \= new tab 2 \\
new \> tab \\
\poptabs % 저장했던 탭 위치 복원
Zero \> One \> Two \> Three
\end{tabbing}
```

```
Zero    One            Two            Three
Zero    One
         Zero        Two            Three
Zero    One        Two
new tab 1... new tab 2
new        tab
Zero    One            Two            Three
```

<tabular>와 <table> 환경도 알아보자.

4.10 Insert▷Cross Reference(교차참조)

텍스트에 라벨을 달고 라벨 위치로 교차참조하기 위해서 `\label`, `\ref`, `\pageref` 명령이 사용된다.^{*23} 라벨은 `prefix:suffix` 형식으로 이름짓는 것이 표준적이다. 여기서 `prefix`는 예를 들면 다음과 같다. 장(chapter) 번호에 `cha`, 수식 번호에 `eq`, 그림에는 `fig`, 절(section)에는 `sec`, 표에 `tab`.

[역주*23] 15 페이지 역주 20에서 `\eqref` 명령에 대해 언급하였다.

절, 표, 그림 등이 놓여 있는 페이지를 참조하려면 `\label`과 `\ref`를 다음 보기와 같이 사용한다.

```
\paragraph{Example.}
\label{par:example}
This paragraph appears in
Section~\ref{par:example} on
page~\pageref{par:example}.
```

Example. This paragraph appears in Section 4.10 on page 20.

물론 `prefix`를 자신의 마음에 드는 것으로 정해도 된다. `enumerate` 리스트에 적용한 예를 보자.

```
\begin{enumerate}
\item{first step: skip to
\ref{item:end}}
↪ \label{item:start}}
\item{another step
↪ (unreferenced)}
\item{end: go back to
\ref{item:start}}
↪ \label{item:end}}
\end{enumerate}
```

1. first step: skip to 3
2. another step (unreferenced)
3. end: go back to 1

4.11 Insert▷Margin Notes(여백문단)

너무 간단하다. `\marginpar{text}`라고 쓰면 된다. 이 기능은 워드 프로세서에서는 잘 찾기 어려운 것이다.

마진노트!

4.12 Insert▷Text Frame(프레임)

프레임으로 둘러싼 텍스트는 문장의 일부를 두드러지게 나타내기 위해 쓰이는데 페이지 사이에서 나누어지지 않는다. 페이지의 고정된 위치에 텍스트 프레임을 두려면 `textpos` 패키지를 이용하면 된다.^{*24} 그 예제를 그림 A.5(부록 A)에 보였다.

[역주*24] 역자는 `abspost` 패키지를 추천한다. 특히 `beamer`에서 `textpos`가 일으키는 문제를 피해갈 수 있다.

페이지 상의 특정 위치에 텍스트를 고정시키는 것이 아니라면 `<minipage>` (미니페이지) 환경을 쓸 수 있다. 지금 보고 있는 텍스트는 `minipage`로 만들어진 것이다. `minipage` 박스는 그 사이에서 페이지가 나누어지지 않는다.

... 이 텍스트는 미니페이지를 박스로 둘러싼 것이다. `<boxedminipage>` 환경이 같은 이름의 패키지에 의하여 제공된다.

`<minipage>` 사용법은 다음과 같은 형식임을 상기하자.

```
\begin{minipage}[position]{width}
...
\end{minipage}
```

<boxedminipage>에서 프레임과 텍스트 사이의 간격은 다음 명령으로 조절한다.

```
\setlength{\fboxsep}{5mm}
```

4.13 Insert▷Image(이미지)

주의: *이미지(image)*와 *그림(figure)*은 다른 것이다. 상세한 것은 4.14절을 보라.

PDF, JPG, PNG, TIFF^{*25} 파일로 된 이미지를 삽입할 수 있다. POSTSCRIPT나 EPS 파일을 포함하려 한다면 아마 컴퓨터를 잘 아는 사람일 것이므로 틀림없이 이 파일을 처리하거나 PDF로 변환하는 방법을 알고 있을 것이다.^{*26}

강력한 패키지 **graphicx**에서 제공하는 `\includegraphics` 명령으로 이미지를 포함할 수 있다.

[역주*25] 저자는 TIFF를 여기서 언급하고 있으나 현재 TeX Live 시스템은 TIFF 포맷의 이미지를 직접 지원하지 않는다. 적절한 변환 도구를 이용하여 PDF나 PNG로 convert 해주어야 한다.



[역주*26] 10페이지 역주 11을 참고하라.

height, width, angle같은 파라미터를 줄 수 있다. width의 설정 단위는 위의 보기에 서처럼 길이를 주어도 되고 텍스트나 행 길이에 상대값으로 부여해도 된다.

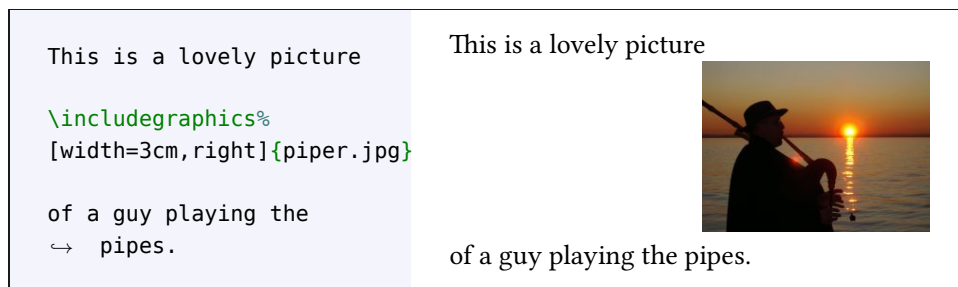
```
width=\textwidth      % 판면의 글줄 너비
width=\linewidth      % 현재 환경의 글줄 너비
width=0.5\linewidth   % linewidth의 절반
```

그림을 가운데 또는 오른쪽 정렬로 배치하려면 **adjustbox** 패키지를 다음에 보인 옵션과 함께 로드하면 된다.

```
\usepackage[export]{adjustbox}
```

adjustbox 패키지에 [export] 옵션을 부여하여 로드하면 `\includegraphics` 명령에 left, right, center, outer, inner와 같은 옵션을 추가적으로 부여할 수 있다. 마지막 두 개의 옵션은 다단 텍스트에서 쓰는 것이다.^{*27}

[역주*27] 저자는 “multi-column text에 적용한다”고 하고 있어서 그대로 옮겼으나 이 부분은 저자의 착각인 듯하다. 정확한 의미는 “twoside(양면) 텍스트에 적용한다”고 해야 한다. 즉 inner라는 것은 책등(spine) 쪽 (오른쪽 페이지의 왼쪽, 왼쪽 페이지의 오른쪽), outer라는 것은 책날개 쪽을 가리키는 것이다.




비트맵 이미지가 PDF 벡터 이미지에 비하여 품질이 같지 않을 것은 말할 필요도 없다. 그리고 그런 이미지들은 실제 출력해보면 상당히 커지는 경향도 있다.

[역자] adjustbox는 매우 다양한 기능을 제공하는 패키지이다. 패키지를 로드할 때 [export] 옵션을 주는 예시를 들고 있는데, 좀더 많은 옵션을 \includegraphics 명령과 함께 쓰게 해주는 [Export]도 있다.

vertical align에 관한 예를 하나 들어 두겠다.

```
\includegraphics[height=12pt,raise=-1ex]{piper.jpg}
```

그림의 수직 위치를  본문에 대하여 조절해야 할 때가 있다.

주의할 점은 raise key를 원래의 \includegraphics 키보다 나중에 선언해야 한다는 것이다.

4.14 Insert▷Figure(삽도)

그림(*figure*) 환경은 이미지(*image*)를 삽입하는 것과 동일하지 않다. 사실 *figure*의 내용물이 이미지일 필요조차 없다. *figure*는 페이지 내의 고정된 특정 위치를 갖는 것이 아니라 떠다니는 것이며 보통 캡션이 붙고 그림 번호에 대하여 교차참조할 수 있다. 이런 일을 하는 것이 <*figure*> 환경이다. 아래에 두 종류 *figure*에 대한 예를 보이겠다.

```
\begin{figure}[htbp]
% [htbp] 플롯이 놓일 위치를 지정
% here, top, bottom, separate
↪ page.
\centering
\texttt{=8-)}
\caption{이 안내서 저자를 나타내는
↪ 스마일리}
\label{fig:mysmiley}
\end{figure}
```

=8-)

그림 1: 이 안내서 저자를 나타내는 스마일리

```
\begin{figure}[htbp]
\centering
\includegraphics%
[width=0.7\textwidth, angle=-90]%
{gnuplot.pdf}
\caption{Gnuplot 그래프}
\label{fig:gnuplot}
\end{figure}
```

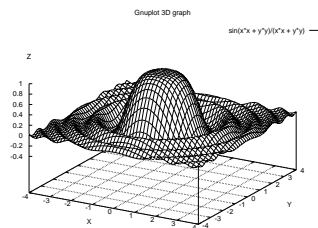


그림 2: Gnuplot 그래프

유념해야 할 것은, *figure*는 반드시 코드를 입력한 바로 그 위치에 나타난다는 보장이 없다는 사실이다. 워드 프로세서와 비교하여 크게 다른 점이 *figure*들이 고정된 위치를 갖지 않는다는 것인데, *figure*가 놓이는 위치는 \LaTeX 이 판단한 가장 최적의 위치로 “떠다니는 (float)” 것이다. 그러므로 *figure*를 지시할 때 “다음 그림”이나 “위의 그림”과 같은 방식으로 언급해서는 안되고 ‘그림~\ref{fig:label}\을 보라’와 같은 식으로 참조해야 한다.

이러한 성질 때문에 figure와 table은 떠다니는 개체(floats)라고 불린다. 플롯 개체를 정확히 특정한 위치에 두고 싶다면 [here](#) 패키지를 사용할 수 있는데 이 패키지는 H(“정확히 이곳(HERE)이라는 뜻이다”) 위치 지정 옵션을 제공한다.^{*28}

[역주*28] memoir/oblivoir에서는 here 패키지를 사용하지 않는다.

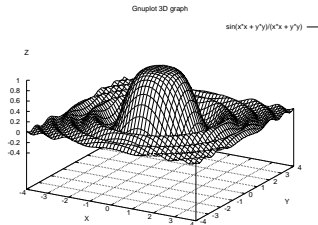
그림을 나란히 놓으려면 subcaption 패키지가 제공하는 subfigure 환경을 사용하여 그림 3과 같은 결과를 얻을 수 있다.^{*29}

[역주*29] subcaption외에도 subfigure, subfloat, subfig 등 subfigure 기능을 제공하는 패키지가 몇 있다. memoir/oblivoir에서는 자체 명령으로도 subcaption을 달 수 있고 이런 패키지를 쓸 수도 있다.

```
\begin{figure}[h]
% first subfigure
\begin{subfigure}{0.5\textwidth} % half figure for 1st subfigure
\includegraphics[width=0.9\linewidth]{piper.jpg}
\caption{Caption of 1st subfigure.}
\label{fig:subfig1}
\end{subfigure}
% second subfigure
\begin{subfigure}{0.5\textwidth}
\includegraphics[width=0.9\linewidth, angle=-90]{gnuplot.pdf}
\caption{Caption of 2nd subfigure.}
\label{fig:subfig2}
\end{subfigure}
% whole figure
\caption{두 개의 subfigure를 가진 figure}
\label{fig:image2}
\end{figure}
```



(a) Caption of 1st subfigure.



(b) Caption of 2nd subfigure.

그림 3: 두 개의 subfigure를 가진 figure

4.14.1 문단을 파고드는 그림

잡지에서 흔히 보는 문단을 파고드는 그림은 wrapfig 패키지를 이용하여 구현할 수 있다.^{*30}

[역주*30] wrapfig2라는 패키지는 원래의 wrapfig을 개선한 것이다. wrapstuff도 일별할 만하다. 만약 그림을 margin에 놓는 것에 관심이 있다면 <marginfigure>라는 환경을 제공하는 패키지가 몇 있다. memoir가 이 환경을 자체 지원하고 있고, sidenotes, sidenotesplus, keyfloat 등이 같은 이름의 환경을 제공한다. 패키지 문서를 참고하라.

If you meet this guy, give him some
 \hookrightarrow money.

```
\begin{wrapfigure}[4]{l}[5pt]{2cm}
{\Huge
~\texttt{=8-)}}
\end{wrapfigure}
The reason may not be clear to you,
but I can assure that your money
will end up in good hands.
I say again, if you meet this guy,
give him some money: he knows how to
use it properly. OK?
```

If you meet this guy, give him some money.
 $=8 -)$ The reason may not be clear
to you, but I can assure that
your money will end up in
good hands. I say again, if
you meet this guy, give him some money:
he knows how to use it properly. OK?

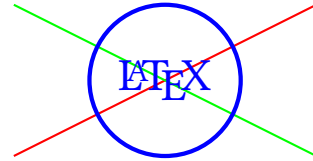
예시된 인자는 그림을 놓기 위하여 폭을 줄여야 할 행의 수, 그림이 놓일 위치(htbp와 같은 위치지정자), 끌어내기 길이, 그림의 폭이다.^{*31}

[역주*31] wrapfig의 위치지정자는 l, r, i, o가 있다. 각각 문단에 대하여 왼쪽, 오른쪽, 안쪽, 바깥쪽이라는 의미이다. htpb와는 달리 단 하나만을 지정해야 한다.

4.15 Insert▷Shapes(도형)

PGF/TikZ는 프로그래밍하듯이 도형을 생성할 수 있는 매우 강력한 패키지이다. 다음은 아주 간단한 예제이다.

```
\begin{tikzpicture}
\draw[red,thick] (-2,-1) -- (2,1);
\draw[green,thick] (-2, 1) -- (2,-1);
\draw[blue,ultra thick] (0, 0) circle
(1cm) node{\huge \LaTeX};
\end{tikzpicture}
```



이런 단순한 것 이외에 실로 엄청난 것을 할 수 있다. TikZ.net을 방문해보라.

만약 WYSIWYG 그림그리기 프로그램이 필요하다면 Inkscape(<https://inkscape.org>)를 추천한다. 이 프로그램은 \LaTeX 을 지원한다. 즉 수식과 텍스트의 입력을 \LaTeX 방식으로 할 수 있다.

Inkscape를 실행하고 적당한 톨로 도형을 그린 다음 메뉴에서 Extensions/Text/Formula (pdflatex)...이나 Extensions/Render/Mathematics/LaTeX (pdflatex)...을 선택하여 그림 4와 같은 방식으로 텍스트를 입력한 다음 Apply를 누르면 \LaTeX 에 의하여 조판된 결과가 삽입된다.

\LaTeX 조판된 텍스트는 그래픽 개체로 포함되는데 필요하다면 그림 5에서 보는 바와 같이 그것을 편집할 수 있다. 완성된 그림은 PDF, PNG 등 \LaTeX 이 지원하는 포맷으로 내보내기하여 활용하면 된다.

\LaTeX 친화적인 출력을 프로그래밍 방식으로 또는 그리기 방식으로 생성하는 그래픽 프로그램이 여럿 있다. 언뜻 머릿속에 떠오르는 것 몇 가지를 들어보면 다음과 같다.

- GLE (Graphics Layout Engine)는 출판물 품질의 그림을 생성하기 위해 설계된 그래픽 스크립팅 언어이다. [...] GLE는 그래프와 도형 속의 텍스트 및 수식 출력을 \LaTeX 에 의존하고 있다. <https://glx.sourceforge.io/>

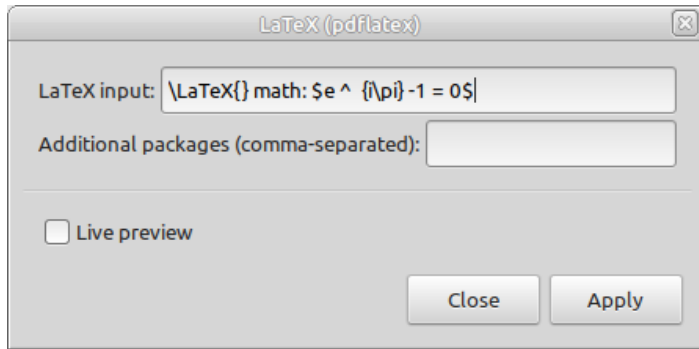


그림 4: \LaTeX 수식 입력

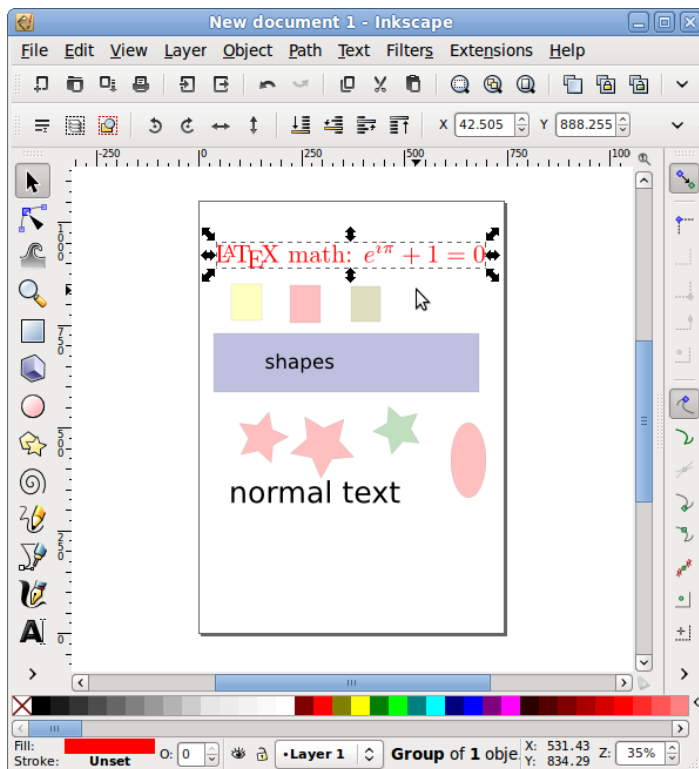


그림 5: \LaTeX 이 생성한 개체의 편집

- Asymptote는 강력한 벡터 그래픽 기술 언어이다. 테크니컬한 도형을 그리기 위한 자연스러운 좌표 기반 프레임웍을 제공한다. 라벨과 수식을 \LaTeX 으로 프린트한다.
<https://asymptote.sourceforge.net/>
- LaTeXDraw는 \LaTeX 을 위한 그림 편집기이다.
<https://latexdraw.sourceforge.net/>
- Graphviz는 간단한 텍스트 언어로 그래프를 기술하며 \LaTeX 에 포함할 다이어그램을 만든다.
<https://graphviz.org>.

이런 패키지를 이용해서 \LaTeX 을 위한 출판 품질의 도형 그림을 얻을 수 있다. 이밖에도 많다. “LaTeX vector graphics”로 웹을 검색해보라.

그릴 수 있는 ‘도형’의 종류는 여러 가지가 있다. TeX Showcase 페이지를 방문해보면 흥미를 자극하는 여러 예를 발견할 수 있을 것이다.
<https://www.tug.org/texshowcase/>.

4.16 Insert▷Page(페이지)

외부 PDF 페이지를 overfull 에러^{*32}를 내지 않고 통째로 삽입하는 것은 pdfpages 패키지를 쓰면 쉽다. 다음과 같은 식으로 사용한다.

[역주*32] overfull은 ‘에러’가 아니라 ‘경고’이다. 그러나 되도록 억제하는 것이 좋다.

```
\includepdf[fitpaper,frame]{pandoc_template.pdf}
```

이 명령은 2.4절에서 보인 Pandoc으로 만들어지는 PDF 문서를 삽입한다. 다음 페이지는 위의 코드를 실행한 결과 생성된 PDF를 \includepdf한 결과이다.


pdfpages는 이밖에도 여러 가지 기능을 갖추고 있으니 패키지 문서를 읽어보는 것이 좋다.

4.17 Insert▷Rule(괘선)

임의의 길이와 두께를 가지는 선은 \rule 명령으로 그린다.

```
This is a page-wide
rule:\\
\rule{\linewidth}{1pt}
but this one is shorter
and thicker:\\
\rule{2cm}{2mm}
```

This is a page-wide rule:

but this one is shorter and thicker:


또하나 흥미로운 것으로, 점선을 그리는 \dotfill 명령이 있다. 유관한 것을 연결할 때 흔히 쓴다.

Total price \dotfill \euro~10

Total price € 10

4.18 Insert▷Hyperlink(하이퍼링크)

hyperref 패키지는 URL 및 외부 참조의 링크를 만들어 PDF가 브라우징 가능하게 만들어 준다. 예를 들면 이 문서는 다음과 같이 선언하였다.

```
\usepackage[colorlinks,
  urlcolor=blue,
  filecolor=magenta,
  linkcolor=darkred,
  hyperfootnotes=false]{hyperref}
```

예를 들어보겠다.

```
\urlstyle{same}
The \hypertarget{ctan}{CTAN} main site
```

```
is \url{https://www.ctan.org}, a.k.a
\href{https://www.ctan.org}{CTAN://}.
```

```
Listen to \href{run:midifile.mid}
{this MIDI file}.
```

```
Click \hyperlink{ctan}{here} to go
back to the top.
```

The CTAN main site is <https://www.ctan.org>, a.k.a [CTAN://](#).

Listen to [this MIDI file](#).

Click [here](#) to go back to the top.

기본적으로 `\url` 명령은 고정폭 글꼴로 인쇄한다. 본문과 같은 글꼴을 쓰고 싶다면 다음 명령을 `\hyperref` 선언 이후에 둔다.^{*33}

```
\urlstyle{same}
```

`\hypertarget`과 `\hyperlink` 명령은 마치 HTML처럼 내부 링크를 생성한다. `\href` 명령은 URL이나 외부 파일로의 링크를 만든다. 주목할 것은 `run:` 파라미터이다. 멀티미디어 플레이어, 오피스 응용 프로그램 등 외부 프로그램을 실행하도록 할 수 있다. 다만 이 기능은 PDF 리더에 따라 지원이 되지 않을 수 있다.

`hyperref` 패키지 문서를 읽어보자. 더 많은 예제와 할 수 있는 일을 알 수 있다.

[역주*33] 원본은 이 선언을 두어서 URL 주소를 본문 글꼴로 식자하고 있는데, 번역본은 그 방침을 따르지 않았다. 다만 위의 예제에서는 이 선언을 활성화하였다.

4.19 Insert▷Comment(주석)

주석문을 만들려면 %를 각 행의 앞에 붙이거나 `comment` 패키지가 제공하는 `\comment`, `\endcomment` 명령과 `comment` 환경을 쓴다.^{*34}

```
This text % boring example
\begin{comment}
what a boring example
\end{comment}
is just an example.
```

This text is just an example.

[역주*34] `memoir/oblivoir`에서는 자체적으로 `<comment>` 환경을 제공한다. `\commentson`, `\commentsoff` 명령으로 이를 보이거나 숨길 수 있다.

PDF 출력물에 팝업 노트식의 주석을 붙이기 위해서는 `pdfcomment` 패키지를 쓰면 된다. 이 안내서에서는 다음과 같이 선언하였고, 아래 PDF 주석을 붙이는 예를 보였다.^{*35}

```
[icon=note,color={1 1 0}]{pdfcomment}
```

```
This text is nothing special
\pdfmargincomment{yup,
↔ definitely.},
but it contains a couple of
\pdfmarkupcomment{nice}%
{simple, more than nice} comments.
```

This text is nothing special, but it contains a couple of **nice** comments.

[역주*35] `pdfcomment`는 \LaTeX 을 지원하지 않지만, 역주 41에서 언급한 `soul` 관련 패키지 `soulpos`를 이 패키지가 활용하기 때문에 `ulem`을 쓰는 문서에서 혹시 약간의 불일치가 있을 수도 있다. `pdfmarginpar`라는 간편한 패키지가 있으나 `pdflatex`만 된다. 그밖에 비슷한 기능을 갖춘 `todonotes`도 살펴볼 수 있다.

PDF 팝업 주석의 경우 PDF 뷰어에 따라 지원하지 않는 경우가 있음을 알아두자.

5 Format 메뉴

일반적으로 문서의 기본 포맷 속성을 설정하기 위해서는 `\documentclass` 명령의 인자로 부여한다. 설정할 수 있는 옵션으로 기본 글꼴 크기(10, 11, 12pt), 용지(a4paper, a5paper, b5paper, letterpaper, legalpaper, executivepaper), 방향(portrait, landscape)같은 것이 있다.

```
\documentclass[a5paper,landscape,12pt]{article}
```

폰트 크기에 대하여 5.2.3절에 더 상세히 설명하겠다.

5.1 Format▷Line Spacing(행 간격)

`setspace` 패키지는 `<singlespace>`, `<onehalfspace>`, `<doublespace>` 환경을 제공한다. 그리고 `\spacing{amount}` 명령(환경)으로 임의의 행 간격 값을 설정할 수 있다.^{*36}

[역자] `\singlespacing`, `\doublespacing`, `\onehalfspacing`은 명령이고, `<singlespace>`, `<doublespace>`, `<onehalfspace>`는 환경이다. 임의의 행간을 설정하는 `<spacing>`은 `...ing`으로 끝나지만 환경이다. 명령어 형태의 매크로는 그 문단이 종료되어야 효과가 나타난다.

[역주*36] `memoir/oblivoir`에서는 `setspace`를 별도로 로드하지 않아도 (몇 글자를 대문자로 바꾼) 같은 환경과 명령을 사용할 수 있다. 예컨대 `\singlespacing`에 대응하는 `\SingleSpacing` 명령이 있고 `<spacing>` 대신 `<Spacing>`을 쓰면 되는 식이다.

```
\begin{spacing}{2.5}
These two lines \\
are crazily spaced!
\end{spacing}
\begin{spacing}{1}
Much better, these lines\\
are spaced normally.
\end{spacing}
```

These two lines

are crazily spaced!

Much better, these lines
are spaced normally.

문서 전체에 걸쳐 행 간격을 달리 설정하려면 `\linespread{factor}` 명령을 프리엠블에 두면 된다. `factor`의 기본값은 1이다.^{*37} 이 값이 커지면 행 사이의 간격도 넓어진다. 배행간은 대략 1.6 정도의 값이 된다.

[역주*37] 한국어 문서는 대체로 행간격을 넓게 잡는 경향이 있다. `koTeX` 계열의 패키지들에 `[hangu1]` 옵션을 주면 대체로 1.3 정도의 값을 적용한다. `oblivoir`의 경우는 한글 워드프로세서의 기본 160%에 가까운 값을 적용하고 있다.

5.2 Format▷Character(글자 모양)

표준적인 글자 모양 속성을 표 4에 열거하였다. 폰트 크기를 바꾸는 명령은 표 5를 보라. 인쇄되는 폰트의 실제 크기는 `documentclass`에 부여된 기본 크기(10, 11 또는 12pt)에 따라 달라진다. 표 6을 보라.

기울인 서체(italics)와 강조체(emphasized)의 차이에 대해 유념해야 한다. 다음 예문을 보면 `\emph`한 부분이 바로 선 서체로 되어 있음을 볼 수 있다. 이와 같이 `\emph`는 서체의 타이포그래피를 지정하는 명령이 아니라 논리적 강조를 나타내는 명령인 것이다.^{*38}

[역주*38] 한글 문서에서 이 논리적 강조를 어떻게 표현할 것인가에 별다른 표준은 없다. 출판 현장에서는 대체로 굵은 서체나 고딕체로 표현하는 것을 선호하는 듯하다. 이 안내서 번역본에서는 의도적으로 우사체로 기울이고 상점을 찍는 (조금 과도한) 표현을 보였는데 우사체는 일반적으로 받아들여지는 방식이라 하기 어렵다. 상점은 그 나름의 역사와 근거가 있으나 최근 서적에서 보기는 어려워졌다. 강조 표현은 문서 설계자에게 맡겨져 있다고 할 수 있다.

글자 속성 명령	환경	예시
<code>\textnormal</code>	<code>textnormal</code>	main document font
<code>\textrm</code>	<code>rmfamily</code>	roman
<code>\textit</code>	<code>itshape</code>	<i>italics</i>
<code>\emph</code>	n/a	<i>emphasis</i>
<code>\textmd</code>	<code>mdseries</code>	medium weight (default)
<code>\textbf</code>	<code>bfseries</code>	boldface
<code>\textup</code>	<code>upshape</code>	upright (default)
<code>\textsl</code>	<code>slshape</code>	<i>slanted</i>
<code>\textsf</code>	<code>sffamily</code>	sans serif
<code>\textsc</code>	<code>scshape</code>	SMALL CAPS
<code>\texttt</code>	<code>ttfamily</code>	typewriter
<code>\underline</code>	<code>underline</code>	<u>underline</u>
<code>\textsuperscript</code>	n/a	this is ^{superscript}
<code>\mathrm</code>	n/a	$x^n + y^n \neq z^n \forall n \neq 2$
<code>\mathbf</code>	n/a	$\mathbf{x}^n + \mathbf{y}^n \neq \mathbf{z}^n \forall n \neq 2$
<code>\mathsf</code>	n/a	$x^n + y^n \neq z^n \forall n \neq 2$
<code>\mathtt</code>	n/a	$x^n + y^n \neq z^n \forall n \neq 2$
<code>\mathit</code>	n/a	$x^n + y^n \neq z^n \forall n \neq 2$
<code>\mathnormal</code>	n/a	$x^n + y^n \neq z^n \forall n \neq 2$
<code>\mathcal</code>	n/a	$\mathcal{X}^{\mathcal{N}} + \mathcal{Y}^{\mathcal{N}} \neq \mathcal{Z}^{\mathcal{N}} \forall \mathcal{N} \neq 2$

표 4: 글자 모양

```
\textit{For example, this
↪ portion of text is typeset
↪ in italics, and \emph{these
↪ words} are emphasised in
↪ upright}.
```

For example, this portion of text is typeset in italics, and these words are emphasised in upright.

또한, 아래첨자는 수식 모드에서만 사용하는 것이 일반적이다. 보통 텍스트에 사용하려면 다음과 같은 트릭을 쓸 수 있다.^{*39}

```
this is
$_{\mbox{\footnotesize%
{subscript}}}$
```

this is _{subscript}

[역주*39] 이러한 트릭을 쓰지 않고 `\textsubscript` 명령을 통하여 일반 텍스트에 아래첨자를 적을 수 있다. `\textsuperscript`도 마찬가지.

5.2.1 화학반응식의 첨자

화학반응식은 수식 모드에서 입력하여 `^`와 `_`로 위첨자와 아래첨자를 식자할 수 있다. 그러나 `mhchem` 패키지를 이용하면 더 간단하다. 숫자는 바로 아래첨자로 들어가며 `^` 부호 다음에 오는 것은 위첨자로 인쇄한다. 반응식은 `\ce` 명령의 인자로 입력한다.

폰트 크기	예시
tiny	sample text
scriptsize	sample text
footnotesize	sample text
small	sample text
normalsize	sample text
large	sample text
Large	sample text
LARGE	sample text
huge	sample text
Huge	sample text

표 5: 폰트 크기

Default font size	10pt	11pt	12pt
tiny	5	6	6
scriptsize	7	8	8
footnotesize	8	9	10
small	9	10	10.95
normalsize	10	10.95	12
large	12	12	14.4
Large	14.4	14.4	17.28
LARGE	17.2	17.28	20.74
huge	20.7	20.74	24.88
Huge	24.8	24.88	24.88

표 6: 포인트 단위로 나타낸 실제 폰트 크기

<pre> \ce{H2O + CO2 -> H2CO3}\ \ce{CaCO3 -> Ca^2+ + \rightarrow CO3^2-}\ \ce{CO3^2- + H2CO3 -> 2 \rightarrow HCO3^-}\ \ce{CaCO3 + H2CO3 -> Ca^2+ + \rightarrow 2 HCO3^-} </pre>	$\text{H}_2\text{O} + \text{CO}_2 \longrightarrow \text{H}_2\text{CO}_3$ $\text{CaCO}_3 \longrightarrow \text{Ca}^{2+} + \text{CO}_3^{2-}$ $\text{CO}_3^{2-} + \text{H}_2\text{CO}_3 \longrightarrow 2 \text{HCO}_3^-$ $\text{CaCO}_3 + \text{H}_2\text{CO}_3 \longrightarrow \text{Ca}^{2+} + 2 \text{HCO}_3^-$
---	--

비슷한 문법을 구현한 [chemformula](#) 패키지도 있다.

5.2.2 밑줄 긋기

밑줄 긋기는 쓰지 않는 것이 일반적이다. 이것은 예전 타자기로 글을 쓰던 시절의 유산으로 그렇게 보기 좋지도 않다. 그럼에도 불구하고 밑줄을 반드시 그어야 한다면 [ulem](#) 패키지를 쓰면 된다.^{*40} 다음 예시에서 이 패키지가 제공하는 밑줄 이외의 몇 가지 스타일을 함께 보았다.

^[역주*40] ko_TeX 패키지 중에서 luatexko 는 자체 ulem 류 명령을 제공한다. X_yTeX 이나 pdf_latex 이라면 ulem 을 별도로 로드한다.

```
\uline{important}
\uuline{urgent}
\uwave{boat}
\sout{wrong}
\xout{removed}
```

important urgent boat ~~wrong~~
~~removed~~

ulem 패키지를 옵션 없이 로드하면 \emph 명령을 재정의하여 이탤릭 대신 밑줄 굵기로 바꾼다. 이것을 피하려면 다음과 같이 옵션을 부여하라.^{*41}

```
\usepackage[normalem]{ulem}
```

[역주*41] ulem과 유사한 기능을 제공하던 soul 패키지는 유니코드나 동아시아 문자를 잘 지원하지 못하는 관계로 특히 xelatex, lualatex에서는 더이상 사용하지 않는다.

5.2.3 Format▷Character Size(큰 글자)

TeX의 표준 사이즈로 충분치 않다면 extsizes 패키지가 도움이 될 것이다. 표준 문서 클래스를 확장하여, 8-12포인트, 14, 17, 20포인트 옵션을 추가로 부여할 수 있게 하였다.

예컨대 본문 17pt article을 작성하려 한다고 생각해보자. 그러면 다음과 같이 시작하면 된다.

```
\documentclass[17pt]{extarticle}
```

큰 글자를 얻는 다른 방법은 type1cm 패키지를 이용하는 것이다. 다음과 같은 명령이 유효해진다.^{*42}

```
\fontsize{72pt}{72pt}\selectfont
No Smoking
```

[역주*42] type1cm는 본문을 type1 CM글꼴로 작성하는 구미어 문헌에서 의미가 있다. 현대적인 오픈타입 폰트를 사용하는 TeX 엔진에서는 이 패키지 없이도 예제가 제시한 명령의 의도한 결과를 얻을 수 있다.

(이 예제의 결과는 이 문서에 표시하기에 너무 크기 때문에 결과 예시를 생략한다.) 두 인자는 각각 폰트 크기와 베이스라인(행송)의 길이이다.

또다른 접근방법으로 다음과 같이 하는 수도 있다.^{*43}

```
\resizebox{!}{1cm}{1-cm tall}
```

1-cm tall

[역주*43] 앞서 소개한 adjustbox에 \resizebox와 마찬가지로 박스 자체를 확대하는 명령이 정의되어 있다. 본문의 예와 같은 역할을 하도록 하는 명령은 \adjustbox{height=1cm}{1-cm tall}이다.

문 단 머리를 큰 글자로 장식하는 lettrine은 lettrine 패키지의 \lettrine 명령을 이용한다. 여러 가지 사용자화가 가능한데 이 문단에서 보인 것은 다음과 같이 기본값을 그대로 쓴 것이다.

```
\lettrine{D}{ropped} capitals at
↪ the
start of a paragraph can be
obtained using the lettrine
↪ package,
```

DROPPED capitals at the start of
a paragraph can be obtained
using the lettrine package,

5.2.4 Format▷Character Font(폰트)

TeX은 원래 독자적인 폰트 시스템을 갖추고 있고 기본 글꼴은 Computer Modern이다. METAFONT 서브시스템을 통해 필요할 때마다 출력용 폰트를 자동으로 생성한다. 이렇게

함으로써 어떤 상황에서도 동일한 출력을 보장하여 훌륭한 결과를 산출할 수 있다. 그러나 우리는 현재, Times, Helvetica, Sans Serif 등 다른 종류의 폰트에 익숙해져 있다.

다행히, \LaTeX 엔진은 POSTSCRIPT Type 1과 오픈타입 OTF 폰트를 활용할 수 있다.^{*44} avant, anagar, bookman, chancery, chanter, courier, helvet, helvetic, ncntrsbk, newcent, palatcm, palatino, pifont, times, utopia, zapfchan 등이 Type 1 폰트를 활용할 때 적용할 수 있는 패키지들이다. `\usepackage{times}`를 프리앰블에 적어넣으면 어떻게 되는지 확인해 보라. 이 문서는 libertine 패키지를 이용하였다.

[역주*44] 폰트 패키지의 유효성은 컴파일하는 엔진에 크게 좌우된다. 이 단락은 주로 `pdflatex`이 컴파일 엔진인 상황에서나 유의미하다. 번역본은 `xelatex`을 이용하여 컴파일하는데 이 때 이런 패키지는 의미가 없으니 주의해서 사용해야 한다. 참고로 libertine 패키지는 `pdflatex`과 `xelatex`에서 모두 유효하게 동작한다.

다만, 수식을 다룰 때에는 Computer Modern 폰트를 적용하는 것이 최선이라는 점에 유의하자. 다른 폰트를 사용하면 수학식이 뭔가 어설픈 결과를 가져온다.

유니코드와 트루타입, 오픈타입 폰트를 지원하는 `xelatex`을 엔진으로 쓴다면 할 수 있는 일이 엄청나게 많아진다. [LaTeX Font Catalogue](#) 웹사이트를 방문하여 어떤 폰트가 있는지 확인해 보자.

앞에서 언급한 패키지들은 문서 전체의 폰트를 설정하는 것이다. 약간의 텍스트에만 특정 폰트를 적용하려 한다면 다음 예제에서 보이는 것처럼 font family를 지정해준다. 흔히 쓰이는 font family를 표 7에 요약해두었다. 몇몇 font shape가 일부에서는 지원되지 않을 수도 있다는 사실을 알아두자.^{*45}

[역주*45] 이 문단에서 말하는 font family, font shape 등은 `pdflatex` 엔진에서 쓰이는 폰트 선택 시스템의 용어이다. 유니코드 폰트를 적용하는 엔진에서는 이 단락의 내용이 적용되지 않는다. 표 7도 마찬가지다.

```
This is Computer Modern Roman,
{\fontfamily{phv}\selectfont
this is Helvetica!}
```

[역자] 역주에서 언급한 대로 위의 예제는 \LaTeX 으로 컴파일하는 번역본에서는 실현되지 않는다. 그 대신 다음 예와 같이 하여야 한다. Latin Modern이란 CM에서 온 오픈타입 글꼴로서 \LaTeX 의 기본 폰트이고 TeX Gyre Heros는 Helvetica에 대응하는 TeX Gyre 글꼴(오픈타입)이다. 폰트 선택을 위해 fontspec을 로드해야 하는데 `pdflatex`을 이 패키지는 지원하지 않는다.

```
\fontspec%
{lmroman10-regular.otf}%
This is Latin Modern
↔ Roman,
{\fontspec%
{texgyreheros-regular.otf}%
this is Heros!}
```

This is Latin Modern Roman, this
is Heros!

5.2.5 Format▷Character Colour(글자 색상)

`color` 또는 `xcolor` 패키지와 관련 명령으로 글자에 색을 입힐 수 있다. 기본적으로 black, white, red, green, blue, cyan, magenta, yellow가 정의되어 있으며 필요하면 색상을 자신이 정의할 수 있다.^{*46}

[역주*46] 미리 정의된 색상명칭을 `xcolor`가 다양하게 제공하고 있으며 `latexcolors`, `ninecolors` 등 색상 정의를 제공하는 패키지들이 있다.

폰트 패밀리	명칭
cmr	Computer Modern Roman
cmss	Computer Modern Sans Serif
cmtt	Computer Modern Typewriter
pag	Avantgarde
pbk	Bookman
phv	Helvetica
pnc	New Century Schoolbook
ppl	Palatino
ptm	Times
pcr	Courier

표 7: 자주 쓰이는 폰트 패밀리 (pdf_latex)

<pre> \textcolor{red}{This is red.}\\ \color{blue} This text is blue!\\ So is this. Let's change.\\ \definecolor{mygreen} {rgb}{0.1,1,0.1} \color{mygreen} This is my shade of green!\\ \color{black} \colorbox{cyan}{A cyan box}\\ \fcolorbox{blue}{green} {A green box in a blue frame} </pre>	<p>This is red.</p> <p>This text is blue!</p> <p>So is this. Let's change.</p> <p>This is my shade of green!</p> <p>A cyan box</p> <p>A green box in a blue frame</p>
--	---

나아가, `\pagecolor`라는 명령이 있는데, 어떤 결과를 가져오게 하는 명령인지 짐작할 수 있겠는가?

5.2.6 Format▷Character Outline(윤곽선 글자)

글자에 색상을 입히는 것만으로 충분치 않다면 `contour` 패키지를 이용하여 윤곽선(외곽선) 글자를 만들어보자. `\contour` 명령은 글자를 16번(또는 지정하는 횟수만큼) 겹쳐 찍어 윤곽선을 만들어낸다.^{*47}

[역주^{*47}] 한글에도 적용된다. [한글예모](#)

<pre> \contourlength{0.5pt} \Large \textcolor{blue}{\contour{red} {Blue text, thin red outline}} \contourlength{2pt} \textcolor{white} ↪ {\contour[32]{blue} {White text, thick blue ↪ outline}} </pre>	<p>Blue text, thin red outline</p> <p>White text, thick blue outline</p>
---	--

5.3 Format▷Paragraph(문단 모양)

문단은 `\`나 빈 줄로 끝나는 텍스트의 일부로서 그 형태는 \LaTeX 이 구성하는 것임을 상기하자.^{*48}

[역주^{*48}] 본문의 취지와는 다르게, 역자는 `\`로 끝나는 것이 ‘문단’이 아니라는 점을 강조해두려 한다. 문단은 빈 줄이나 `\par`로 끝나는 것을 말한다. `\`는 단지 한 문단 내에서의 강제 줄바꿈이다.

환경	설명
abstract	요약문
array	수학식의 배열
center	가운데 정렬
description	설명형 리스트
displaymath	별행 문단형 (displayed) 수식
document	문서 전체를 둘러싸는 환경
enumerate	번호부 리스트
equation	별행 문단형 (displayed) 수학적 (수식번호 동반)
figure	떠다니는 그림
flushleft	왼쪽 정렬
flushright	오른쪽 정렬
itemize	부호부 리스트
letter	편지 형식
list	기저 리스트 환경
math	행중 (In-line) 수식
minipage	미니페이지
picture	텍스트, 화살표, 패선, 원 등의 도형
quotation	여러 문단 인용문 (첫 줄 들여쓰기 있음)
quote	한 문단 인용문 (첫 줄 들여쓰기 없음)
tabbing	탭 스톱을 이용한 정렬
table	떠다니는 표
tabular	행과 열로 정렬된 텍스트
thebibliography	문헌 목록 리스트
theorem	정리, 명제 등
titlepage	표제지 작성
verbatim	입력을 그대로 보이기
verse	시와 운문

표 8: 표준 \LaTeX 환경

텍스트의 일부에 특정한 속성, 예컨대 정렬 방식이나 글꼴 모양 등을 부여하기 위해 \LaTeX 은 *환경 (environment)*을 이용한다. 이것은 마치 마우스로 텍스트 일부를 선택하여 메뉴로부터 특정 속성을 선택하거나 버튼을 클릭하는 것과 유사하다. 중괄호로 텍스트 일부를 둘러싸는 것도 또한가지 방법이다.

환경은 다음과 같은 형식으로 쓴다.

```
\begin{environment}
...text goes here...
\end{environment}
```

예를 들어보자. 한 문단을 가운데 정렬하기로 하면 `<center>` 환경을 적용한다.

```
\begin{center}
this text is centered
\end{center}
```

this text is centered

표준 환경을 표 8에 요약하였다. 아래 그 사용법을 몇 가지 보이겠다.

5.3.1 Paragraph▷Horizontal Alignment(정렬)

기본적으로 문단은 양끝맞춤으로 정렬한다. 왼쪽 정렬이나 오른쪽 정렬 또는 가운데 정렬은 `<flushleft>`, `<flushright>`, `<center>` 환경 안에 텍스트를 넣는다. 명령 형식인 `\raggedright`, `\raggedleft`, `\centering`이 이 환경과 각각 유사하게 정렬하는 결과를 만들지만, 새로운 문단이 시작하기 전까지는 효과가 나타나지 않는다.

5.3.2 Paragraph▷Vertical Alignment(문단 사이의 정렬)

문단을 구분하는 \TeX 의 방식에 워드 프로세서 사용자들이 당황하는 경우가 종종 있다. 빈 줄이나 공백이 여러 개 와도 하나의 빈 줄, 하나의 공백으로 간주한다. 따라서 빈 줄을 여러 개 둔다고 해서 문단 사이가 더 벌어지는 것이 아니라는 말이다. `\smallskip`, `\medskip`, `\bigskip`이라는 명령을 문단 사이에 명시적으로 적어넣어야 문단과 문단 사이의 거리를 얻을 수 있다.

이보다 더 간격을 벌리고 싶다면 `\vskip`을 다음 보기와 같이 적어넣으면 된다.^{*49}

<pre>These paragraphs will be separated by 1.3 cm:\vskip 1.3cm there is a 1.3 cm gap above me.</pre>	<p>These paragraphs will be separated by 1.3 cm:</p> <p>there is a 1.3 cm gap above me.</p>
--	---

[역주*49] `\vskip`은 \TeX 명령이고 `\vspace`는 \LaTeX 명령이다. 되도록 `\vspace`를 쓰는 쪽이 의도치 않은 에러를 방지할 수 있다. `\vskip`은 예시에서와 같이 길잇값을 그 뒤에 적고, `\vspace`는 길이를 중괄호로 둘러싸서 인자로 전달한다.

`\vskip`은 오로지 문단 사이에서만 동작한다. 새로운 페이지의 상단에(즉 이전의 문단이 없는 상태에서) 예컨대 1.5cm의 공백을 두고 문단을 시작하고 싶다면 어떻게 해야 할까? `\null`을 (없는 문단 대신) ‘표지’로 삼아 간격을 설정해야 한다.^{*50}

<pre>\null \vskip 1.3 cm This text comes after 1.3 cm...</pre>	<p>This text comes after 1.3 cm...</p>
--	--

[역주*50] \TeX 의 `\vspace` 명령에 별표를 붙여서 `\vspace*` 형식으로 쓰면 명령이 주어진 위치 이전(및 이후)에 기존 문단이 없어도 수직 간격을 벌리기 때문에 본문에서 설명하고 있는 것과 같은 효과를 얻을 수 있다.

끝으로, `\vfill` 명령이 있다. 두 문단 사이를 빈 공간으로 채우는 데 사용한다. 즉 두 번째 문단은 항상 그 페이지의 바닥에 오게 된다. 다음 예를 참고하라.

<pre>This appears at the top of the page{\ldots} \vfill {\ldots}and this at the ↪ bottom.</pre>	<p>This appears at the top of the page...</p> <p>...and this at the bottom.</p>
---	---

5.3.3 Paragraph▷Margins(문단 폭)

2.5에서 본 바, 문단 폭은 문서 전체에 걸쳐 적용되는 것이다. 한 섹션에만 문단 폭을 변경하는 것은 일반적으로 허용되지 않는다. 한 문단의 문단 폭을 별도로 설정하려 한다면 다음 예시에서 보이는 것과 같이 새로운 환경을 작성해야 한다.

```
\newenvironment{margins}[2]
{
  \begin{list}{} {
    \setlength{\leftmargin}{#1}
    \setlength{\rightmargin}{#2}
  } \item }
{\end{list}}
```

이 새로이 정의한 환경을 다음과 같이 사용한다. ^{*51}

```
As you can see, this paragraph
has normal margins.
\begin{margins}{0.5cm}{1cm}
But please note that this
paragraph has custom margins.
\end{margins}
```

As you can see, this paragraph has normal margins.

But please note that this paragraph has custom margins.

[역주*51] 이 안내서에서는 새로운 환경을 정의하여 쓰고 있는데, memoir에 <adjustwidth>라는 환경이 이 기능과 거의 유사한 것을 제공한다. memoir의 것은 changepage 패키지(예전 이름은 chngpage)의 것과 동일하다.

5.3.4 Paragraph▷Indentation(들여쓰기)

문단 첫 줄 들여쓰기의 크기를 정하려면 \parindent라는 길이 변수의 값을 재정의한다. 다음 보기는 들여쓰기를 1cm로 하는 것이다.

```
\setlength{\parindent}{1cm}
```

\indent와 \noindent 명령은 그 다음에 오는 문단의 들여쓰기를 제어한다. 끝으로, \parskip 길이 파라미터로 문단 사이의 간격을 지정해줄 수 있다. ^{*52}

```
\setlength{\parskip}{3pt}
```

[역주*52] 따라서, 요즘 유행하는 모던한 문서처럼 문단 사이에 일정한 간격을 주고 싶을 때, 강제로 빈 줄을 하나씩 다 집어넣는 것이 아니라는 데 주의하라. \parskip을 바꾸려 한다면 parskip 패키지를 사용하는 것이 좋다.

5.3.5 Paragraph▷Border and Shade(프레임과 음영)

문단이나 단어에 프레임을 치려 한다면 framed 패키지를 쓴다. 또는 \parbox 명령을 응용하면 되는데 이 때는 calc 패키지가 필요하다. ^{*53}

쉽게 framed를 이용하는 예를 보이자.

```
\setlength{\FrameRule}{2pt}
\setlength{\FrameSep}{5pt}
\begin{framed}
  this is a framed paragraph!
\end{framed}
\definecolor{shadecolor}{rgb}
{0.9,0.8,1}
\begin{shaded}
  this is a shaded paragraph,
  do you like it?
\end{shaded}
```

this is a framed paragraph!

this is a shaded paragraph, do you like it?

[역주*53] memoir/oblivoir에서는 framed 패키지의 주요 기능을 자체 구현하고 있다. 한편, 프레임친 텍스트 구현에 있어 살펴볼 만한 패키지로 adjustbox, efbox 외에 tcolorbox를 반드시 언급해두려 한다. 이 엄청난 패키지는 팬시한 박스를 그리는 문제에 있어서 필요한 거의 대부분의 기능을 제공한다.

그리고 `boxedminipage` 패키지가 있는데, 같은 이름의 환경을 쓸 수 있게 한다. 조금 힌트를 주자면 다음과 같은 명령을 내리는 것은 사실상 `<boxedminipage>` 환경을 쓰는 것과 동일하다.

```
\framebox{
  \begin{minipage}[c]{\linewidth}
    text to be framed
  \end{minipage}
}
```

`\width` 매크로는 인자로 주어지는 텍스트의 내용을 담은 `minipage`의 길잇값을 가지고 있다. 물론 원한다면 특정한 길이를 지정해도 상관없다.

주어진 텍스트에 맞는 길이의 프레임을 그려보자.

<pre>this is a \framebox[\width]{framed} word</pre>	this is a framed word
---	---

길잇값을 수정하여 프레임의 폭을 바꿀 수 있다.

<pre>this is another \framebox[2\width][r]{framed} word</pre>	this is another framed word
---	---

두 번째 옵션 인자는 프레임 안의 내용이 놓일 때의 정렬 방식을 지정한다. 이 보기에서는 오른쪽(r) 정렬이다.

5.3.6 Paragraph▷Colour(문단 색상)

문단에 프레임을 쳤으니 이제 색상도 설정하고 싶다. 이렇게 하자.

<pre>\colorbox{yellow}{ \begin{minipage} {0.8\linewidth} I am a minipage, my ↳ colour is yellow! \end{minipage} }</pre>	<div style="background-color: yellow; border: 1px solid black; padding: 5px;"> I am a minipage, my colour is yellow! </div>
---	---

예시에서 보였듯이 `minipage`와 그 색상은 행 너비의 80%만 적용되도록 설정하였다. 색상에 대하여 더 자세한 사항은 5.2.5절을 참고하라.

5.3.7 Format▷Columns(다단)

`\twocolumn`과 `\onecolumn` 명령은 새로운 페이지를 열고 2단 또는 1단 조판을 시작한다.

`\documentclass`의 옵션 파라미터로 지정할 수도 있다. 이것으로 충분치 않다면 `multicol` 패키지가 제공하는 `<multicols>` 환경이 있다. 이를 이용하여 이 절의 내용을 이단으로 조판하고자 했다면 다음처럼 하였을 것이다.^{*54}

^[역주*54] 원문은 패키지 이름을 `multicols`라고 하고 있는데, 실수로 보인다. 올바르게 역자가 수정하였다. `multicol`의 장점은 반드시 새로운 페이지를 시작하지 않아도 된다는 것과 n 단 텍스트를 조판할 수 있다는 점이다.

```

\columnseprule=1pt
\begin{multicols}{2}[\subsection{\entry{Format}{Columns}}]
The commands \cmd{twocolumn} ...
\end{multicols}

```

단 사이의 간격은 `\columnsep` 길이 변수로 제어한다. `\columnseprule` 매크로는 단 사이에 그어지는 선의 굵기를 나타낸다.^{*55} 대괄호로 묶은 옵션 인자 텍스트는 다단 환경을 시작하기 전에 해당 환경의 영향을 받지 않도록 조판하는 부분이다.

[역주*55] 선을 긋지 않으려면 이 값을 0pt로 한다.

6 Table 메뉴

지옥(그런가?)에 오신 것을 환영한다. 이 난감한 주제를 얘기해보자.

*이미지(image)*가 *figure*와는 다른 것과 마찬가지로, `<tabular>` 환경과 표(*table*)도 서로 다르다. \LaTeX 의 table이란 것은 떠다니는 개체(4.14절에서 설명)로서 한 페이지를 넘여가 지 않는 것이어야 한다. 캡션을 달 수 있고 대체로 `<tabular>` 환경을 그 안에 포함한다. `<tabular>`는 특별한 문법으로 입력해야 하는데 이후 설명하겠다.

기본 \LaTeX tabular 개체는 다음과 같이 정의한다.

```

\begin{tabular}{|l|c r|}
% 좌측, 중앙, 우측 정렬
% 컬럼은 3, 세로줄
% 세로줄은 표를 미워보이게 한다
\hline
1 & 2 & 3 \\\ % 행의 끝
\hline % 수평 패선
a & & c \\\ % 두번째 컬럼 빔
one & two & three \\\
\hline
\end{tabular}

```

1	2	3
a		c
one	two	three

여기서 수직선, 정렬, 컬럼의 수를 지정한 것을 볼 수 있다. tabular의 한 행은 &에 의하여 분리된 컬럼들로 이루어진다.

booktabs 패키지를 이용하면 세로줄이 없는 더 세련된 모양을 갖춘 tabular를 그릴 수 있다.^{*56}

[역주*56] memoir/oblivoir는 booktabs를 기본으로 불러들인다.

```

\begin{tabular}{lcr}
\toprule
1 & 2 & 3 \\\
\midrule
a & & c \\\
one & two & three \\\
\bottomrule
\end{tabular}

```

1	2	3
a		c
one	two	three

다음에 보이는 엄청난 예제는 tabular 개체를 포매팅하는 방법을 설명한다.

```
% 4열 표; 왼쪽/중앙/오른쪽 정렬; 고정폭
\begin{tabular}{lcrp{4cm}}
\hline % 수평패선
\textbf{Left} & \textbf{Centre} & \textbf{Right} & \\
\textbf{4 cm} & \hline
row 1, col 1 & row 1, col 2 & row 1, col 3 & row 1, col 4 \\
\cline{1-2} % 1-2 컬럼에 패선
row 2, col 1 & row 2, col 2 & row 2, col 3 & row 2, col 4 \\
\cline{1-2}
\multicolumn{2}{|c|}{spanning two columns} & row 3, col 3 & \\
row 3, col 4 \\
\cline{1-3}
row 4, col 1 & row 4, col 2 & row 4, col 3 & ~ \hfill right \\
% ~에 의한 강제간격
row 5, col 1 & row 5, col 2 & row 5, col 3 & left \hfill ~ \\
row 5, col 1 & row 5, col 2 & row 5, col 3 & \\
~ \hfill centre \hfill ~ \\
\hline
\end{tabular}
```

Left	Centre	Right	4 cm
row 1, col 1	row 1, col 2	row 1, col 3	row 1, col 4
row 2, col 1	row 2, col 2	row 2, col 3	row 2, col 4
spanning two columns		row 3, col 3	row 3, col 4
row 4, col 1	row 4, col 2	row 4, col 3	right
row 5, col 1	row 5, col 2	row 5, col 3	left
row 5, col 1	row 5, col 2	row 5, col 3	centre

tabularray

이 소절은 역자가 추가한다. 도표(tabular)는 오랫동안 \LaTeX 사용자를 괴롭혀온 문제였다. 다양한 요구와 거기에 대응하는 솔루션, 패키지들이 나옴으로써 대부분의 요구가 실현가능하게 되었으나 너무 많은 해결책 때문에 오히려 무엇이 어디 있는지 알기 어려운 상태가 되었다.

tabularray는 도표를 그리는 매우 강력하고 종합적인 패키지이다. 이 안내서를 포함하여 도표에 대하여 설명하는 문서에서 제기하는 거의 대부분의 기능을 `key=value` 문법을 이용하여 구현할 수 있다. 방대한 매뉴얼에서 원하는 것을 찾을 수 있을 것이다.

위에 제시된 표를 tabularray의 `<tblr>`로 그리는 예를 보이겠다.

```
\begin{tblr}{
^^Icolspec={Q[l]Q[c]Q[r]Q[4cm,l]},
^^Ihline{1,Z}={.8pt}, hline{2}={.6pt},
^^Ivline{3}={.4pt},
^^Ivline{3}={4}{dashed}
}
\SetRow{font=\bfseries,fg=olive5} Left & Center & Right & 4cm \\
row 1, col 1 & row 1, col 2 & row 1, col 3 & row 1, col 4 \\
↔ \cline{1-2}
row 2, col 1 & row 2, col 2 & row 2, col 3 & row 2, col 4 \\
\cline[dashed]{1-2}\rline[dashed]{4}
```

```

\SetCell[c=2]{bg=gray!30,c} spanning two columns & row 3, col 3 &
↪ row 3, col 4\\ \cline[dashed]{1-2}\cline{3}
row 4, col 1 & row 4, col 2 & row 4, col 3 & \SetCell{r} right\\
row 5, col 1 & row 5, col 2 & row 5, col 3 & left \\
row 5, col 1 & row 5, col 2 & row 5, col 3 & \SetCell{c} centre\\
\end{tblr}

```

Left	Center	Right	4cm
row 1, col 1	row 1, col 2	row 1, col 3	row 1, col 4
row 2, col 1	row 2, col 2	row 2, col 3	row 2, col 4
spanning two columns		row 3, col 4	
row 4, col 1	row 4, col 2	row 4, col 3	right
row 5, col 1	row 5, col 2	row 5, col 3	left
row 5, col 1	row 5, col 2	row 5, col 3	centre

tabular가 너무 길어서 한 페이지 안에 들어가지 않는 때가 있다. [longtable](#) 패키지를 이용하여 여러 페이지에 걸치는 tabular를 작성하는 것이 한 가지 방법이다. 한편 [rotating](#) 패키지는 `<sideways>`라는 환경을 제공하는데 이를 이용하면 tabular 전체(또는 cell 하나)를 세로로 놓거나 주어진 각도만큼 회전시킬 수 있다.

[tabularx](#) 패키지는 표의 너비를 고정폭으로 지정할 수 있고 X 컬럼 지시자를 이용하여 컬럼을 전체 표에 대하여 상대적인 길이로 넓히거나 좁힐 수 있게 한다.

다음은 예시이다.

```

\begin{sideways}
\begin{tabularx}{7.5cm}
↪ {\l|X|X|}
\hline
\textbf{normal} &
↪ \textbf{tilted} &
\textbf{wider}\\
\hline
normal & \rotatebox{30}%
{I'm tilted!} &
I'm wider\\
\hline
\end{tabularx}
\end{sideways}

```

	wider	I'm wider
tilted	I'm tilted!	
normal	normal	

그밖에도 [longtable](#), [supertabular](#), [xtable](#) 같은 유용한 패키지들이 있다.

표 안에서 색상을 쓰려면 [colortbl](#) 패키지를 이용한다.

```

Colour by row:\\vskip 2mm
\begin{tabular}{lcr}
\rowcolor{cyan}
one & two & three\\
\rowcolor{green}
one & two & three\\
\rowcolor{yellow}
one & two & three\\
\end{tabular}

```

Colour by row:

one	two	three
one	two	three
one	two	three

```

Colour by column:\\vskip 2mm
\begin{tabular}
{>\columncolor{cyan}}l
>\color{red}
\columncolor{green}}c
>\columncolor{yellow}}r}
one & two & three\\
one & two & three\\
one & two & three\\
\end{tabular}

```

Colour by column:

one	two	three
one	two	three
one	two	three

6.1 Table▷Line Spacing(표의 행 간격)

테이블의 행 간격은 텍스트의 높이에 맞추어서 조절된다. 한 행이 시작하기 전에 추가적인 간격을 주려 한다면 가로(length)가 0pt이고 일정한 높이(height)를 갖는 보이지 않는 패선 \rule을 그리는 트릭이 있다. 행 이후에 간격을 주려면 \\에 옵션 인자로 길이를 지정한다. 다음 예를 보자.

```

\begin{tabular}{lll}
one & two & three\\
0.3 centimeters & & \\
↪ \textbf{after} & & \\
this line\\[0.3cm]
one & two & three\\
one & two & three\\
\rule{0pt}{1.2cm}1.2
↪ centimeters & & \\
\textbf{before} & & this
↪ line\\
\end{tabular}

```

one	two	three
0.3 centimeters	after	this line
one	two	three
one	two	three
1.2 centimeters	before	this line

6.2 Table▷Aligning Numbers(수치 정렬)

tabular를 그릴 때 특별한 경우로서 수치 값의 소수점 위치를 기준으로 정렬해야 할 때가 있다.

가장 쉬운 방법은 @ 컬럼 지시자를 이용하는 것이다. 숫자만을 포함하고 있는 표에서는 제법 쓸 만하다. 컬럼 분리자 &가 소수점으로 바뀐다.

```

\begin{tabular}{r@{.}l}
3&14159\\
1&61803\\
1&41421\\
100&00000
\end{tabular}

```

```

3.14159
1.61803
1.41421
100.00000

```

한편, `dcolum` 패키지를 이용하면 D 컬럼 지시자를 추가로 쓸 수 있게 되는데, 이 지시자는 세 개의 인자를 갖는다. 각각 \LaTeX 소스에 표기된 소수점, 출력으로 나타나는 소수점 (보통은 둘 다 `.` 이다), 그리고 소수점 아래(오른쪽) 자릿수이다. 세 번째 인자를 점(dot)으로 구분하여 입력하면 소수점 왼쪽 자릿수와 오른쪽 자릿수를 나타내게 할 수 있다. 만약 값이 -1 이면 컬럼의 내용은 소수점을 기준으로 가운데 정렬로 나타난다.

이 패키지는 표의 모든 입력이 모두 수학 모드라고 간주하므로, 첫 줄에 텍스트 헤더를 쓸 때는 `\mbox` 안에 넣어서 텍스트 모드로 찍히도록 해야 한다.^{*57}

[역주*57] `\mbox` 대신 `amsmath`의 `\text`를 쓸 수 있다.

```

\begin{tabular}{D{.}{,}{4.2}%
D{.}{.}{5}D{.}{.}{-1}}
\toprule
\mbox{One} & \mbox{Two} &
\mbox{Three}\\
10.33 & 10.33 & 10.33\\
1000 & 1000 & 1000\\
5.1 & 5.1 & 5.1\\
3.14 & 3.14159 & 3.14159\\
\bottomrule
\end{tabular}

```

One	Two	Three
10,33	10.33	10.33
1000	1000	1000
5,1	5.1	5.1
3,14	3.14159	3.14159

6.3 diagbox(셀의 대각선)

`diagbox` 패키지는 `\diagbox` 명령을 제공한다.^{*58}

[역주*58] 예전에 `slashbox` 패키지로 작성한 소스와의 호환성을 위하여 `diagbox` 패키지는 `slashbox`의 몇 가지 명령에 대한 지원을 하고 있다. 그러므로 없어진 `slashbox`를 찾아다닐 필요가 없다.

```

\begin{tabular}{|l|l|l|}
\toprule
\diagbox{Lesson}{Date}
↔ &
Monday & Tuesday\\
\midrule
Stratigraphy & room A
↔ & room A\\
Chemistry & room B &
↔ Lab  $\alpha$ \\
Physics & room C & Lab
↔  $\delta$ \\
\bottomrule
\end{tabular}

```

Lesson \ Date	Monday	Tuesday
Stratigraphy	room A	room A
Chemistry	room B	Lab α
Physics	room C	Lab δ

6.4 L^AT_EX 표 생성기

솔직히 말해서 tabular의 내용을 하나씩 채워넣는 것은 귀찮은 일이다. 다행히 시각적으로 만든 tabular 내용을 L^AT_EX 코드로 생성해내는 웹사이트가 있다.

- <https://www.tablesgenerator.com/>
- <https://www.latex-tables.com/>
- <https://tableconvert.com/latex-generator>
- <https://products.aspose.app/tex/latex-table-generator>

6.5 L^AT_EX 테이블로 데이터 가져오기

우리 모두에게 데이터 파일은 일상 업무를 수행하는 데 꼭 필요하다. 데이터 파일은 대부분 숫자로 채워진 컬럼을 구분한 단순 ASCII 파일일 때도 있지만, 스프레드시트 형식이기도 하다. 거의 대부분의 스프레드시트 프로그램은 ASCII 베이스의 .csv 파일 포맷으로 내보내기하는 것이 가능한데, 컬럼 구분자는 ‘;’로 하는 것이 좋다.^{*59}

데이터 파일의 내용을 L^AT_EX 표 형식으로 변환하여 입력하는 것은 지겨운 일이다. 아래 소개하는 프로그램은 UNIX 용 sh 스크립트인데 데이터 파일을 임의의 컬럼 수에 맞추어 L^AT_EX 코드로 변환한다. .csv 파일에 대해서도 동작할 것이다.

[역주*59] 컬럼 구분자를 세미콜론이 아닌 TAB 문자나 기타 문자를 써도 좋지만 (CSV라는 이름의 의미와는 달리) 침표(.)는 그다지 권장하지 않는데, 그 이유는 각 컬럼의 셀에 침표가 들어가는 텍스트가 존재할 수 있기 때문이다.

```
#!/bin/sh

# dat2tex.sh: converts tabular data to a tabular environment

if [ $# != 1 ]; then
    echo "Usage: $0 <datafile>"
    exit 1
fi

# is this a csv file?
grep ";" $1 > /dev/null
if [ $? = 0 ]; then
    AWK="awk -F;"
else
    AWK=awk
fi

# ok awk, make my day
$AWK '{if (1 == FNR) { \
    printf "\\begin{tabular}{"; \
    for (i = 1; i <= NF; i++) {printf "l"}; \
    printf "}\n"
}
for (i = 1; i < NF; i++) \
    {printf $i" & "} printf $NF"\\\\ \\n"} \
END {printf "\\end{tabular}\\n"}' $1

# end of dat2tex
```

7 Tools 메뉴

7.1 Tools▷Mail Merges(메일 머지)

이 유용하고 시간을 절약하게 해주는 기능은 [textmerg](#) 패키지에 구현되어 있다. 동일한 내용의 문서에서 이름과 성, 호칭만을 바꾸어주는 간단한 경우를 생각해보자. 그 이외의 내용은 동일하다.

문서에서 달라지는 부분에 해당하는 세 개의 필드(*field*)를 정의할 것이다. 각각 \Name, \Surname, \Title이라고 하자. 이에 해당하는 값은 외부 파일인 data.dat로부터 가져온다.

```
\documentclass{article}
\usepackage{textmerg}
\begin{document}
% let's declare the variable fields:
% \Void is for empty lines
\Fields{\Name\Surname\Title-\Void}
\Merge{data.dat}{%
Dear \Title{} \Surname,\\
may I call you \Name?\\
Yours,\\
\hspace{3cm}Guido\clearpage}
\end{document}
```

네 번째 필드인 \Void는 실제 필요한 것은 아니지만 예시를 위하여 두었다. 그 앞에 마이너스 기호가 붙어 있는데 이것은 데이터 파일에서 비어 있을 수 있음을 나타낸다. 각 레코드의 구분은 빈 줄로 한다.

data.dat 파일의 내용은 이런 식으로 되어 있다.

```
Guido
Gonzato
Dr.

Francesco
Mulargia
Prof.

Marie
Curie
Mme
```

이것으로 완성이다. 출력 결과물은 데이터를 포함하여 작성될 것이고 각 수신인별로 한 페이지씩 생성할 것이다.

7.2 Tools▷Labels(라벨 작성)

메일 머지가 쉽다면 라벨 만들기는 더 간단하다. 3×8 스티커 라벨 용지 위에 동일한 라벨 20개를 만든다고 해보자. 사용할 패키지는, 짐작하겠지만 [labels](#)라는 것이다. 다음 예시에서 10개는 그냥 라벨로, 10개는 박스친 라벨로 작성하였다.

```

\documentclass[a4paper,12pt]{article}
\usepackage{labels}
\LabelCols=3      % n. of columns of labels
\LabelRows=8      % n. of rows of labels
\LeftBorder=8mm   % borders of each label
\RightBorder=8mm
\TopBorder=5mm
\BottomBorder=5mm
\LabelGridtrue    % show the grid
\numberoflabels=10 % number of labels of each type to print
% the text of the label is specified by
% the \addresslabel[]{} macro:
\begin{document}
  \addresslabel[\large] % optional arguments
  {\textbf{Guido Gonzato}, Ph.D.\}
  \textsl{GNU/Linux Sysadmin}}
  % now on to the boxed labels
  \boxedaddresslabel[\fboxsep=4mm\fboxrule=1mm]
  {\textbf{Guido Gonzato}, Ph.D.\}
  \textsl{GNU/Linux Sysadmin}}
\end{document}

```

라벨이 서로 다른 주소를 포함하도록 작성하려면 외부의 주소록 데이터 파일로부터 각 주소를 불러오도록 하거나 메인 파일에 주소를 적어넣어야 할 것이다.

```

\documentclass[a4paper,12pt]{article}
\usepackage{labels}
\LabelCols=3
\LabelRows=8
\LeftBorder=3mm
\RightBorder=3mm
\TopBorder=8mm
\BottomBorder=8mm
\LabelGridtrue
\begin{document}
% use either this environment:
\begin{labels}
  1${st}$ name
  1${st}$ address
  1${st}$ city, state, zipcode

  2${nd}$ name
  2${nd}$ address
  2${nd}$ city, state, zipcode

  3${rd}$ name
  3${rd}$ address
  3${rd}$ city, state, zipcode
\end{labels}
% or an external file containing exactly the same text:
% \labelfile{addresses.dat}
\end{document}

```

textmerg와 labels를 결합하여 활용하는 것도 시도해보기 바란다.

7.3 Tools▷Default Language(다국어)

TeX의 기본 언어는 영어이다. 그러나 다른 언어도 지원한다. 여기서 언어 지원이라 함은 ‘chapter’나 ‘index’와 같은 용어를 각국어에 맞게 생성하는 것, 올바른 하이프네이션 규칙, 그리고 ‘c’나 ‘e’와 같은 글자들을 키보드로부터(보통은 \c c나 \e e와 같이 입력하는 것) 입력받는 것등을 의미한다.

TeX 배포판에는 language.dat라는 파일이 (\$TEXMF/tex/generic/config/language.dat) 포함되어 있는데 여기에 각 언어 목록이 있다. 이 파일을 편집하여*60 사용자가 원하는 하이픈 규칙을 가지는 언어를 선택할 수 있다.

영어 사용자가 아니라면 babel 패키지를 다음 예와 같이 사용한다.

```
\usepackage[italian,english]{babel}
```

[역주*60] 이 파일을 사용자가 편집하면 안 된다. 이 언어 데이터 파일은 TeX Live가 생성하는 것이므로 그 상태 그대로 유지되어야 한다. 사용자가 이 파일을 편집할 수 있다는 것처럼 보일 수 있어서 역자 주의를 붙인다. 사용자 수준에서 언어 선택은 babel이나 언어 지원 패키지를 이용하는 것이다.



babel은 일부 문자를 활성화된 언어에 적합하게 바꾸는 경우가 있다. 만약 문자가 원하는 모양으로 나타나지 않는다면 \charXX 방식으로 입력해보자.

또한 부호붙은 문자나 일반 유니코드 문자를 쓰려면 에디터가 UTF-8를 이해하도록 해야 할 필요가 있다. 그리고 2018년 이전에는 inputenc와 fontenc 패키지가 필요했다.

```
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
```

2018년 이후, TeX의 기본 문자 인코딩은 UTF-8이 되었다.

한국어 문서

이 문단은 역자가 추가한다. 한국어를 한글로 적는 문서를 작성하는 방법의 표준은 koTeX 패키지(군)를 이용하는 것이다. pdflatex, xelatex, lualatex 각 엔진에 대하여 한글 처리의 일관성을 제공하고 한글 양식도 구현한다.

```
\usepackage{kotex}
```

koTeX 없이도 한글을 표시하는 것은 문제가 없다. 다양한 엔진에 대하여 다양한 방법이 개발되어 있다. 예를 들면 pdflatex에서의 cjk-ko, xelatex에서의 polyglossia 등이 있으며, 특히 lualatex 하에서 babel 패키지의 사용 가능성도 높아져 가고 있다.

7.4 Tools▷Hyphenation(하이프네이션)

대체로 TeX이 단어의 하이프네이션을 잘 처리하는 편이지만 때로 수작업으로 하이픈 처리하는 것이 필요할 수 있다. 수작업 하이픈을 위해서 잘라질 수 있는 위치에 \-를 지시한다. 더 나은 방법으로 하이픈 규칙을 다음과 같이 선언해두는 것도 가능하다.

```
\hyphenation{ge-o-phy-sics ge-o-lo-gy earth}
```

위의 하이픈 규칙 선언 예시에 의하면 ‘earth’라는 단어는 어디에서도 하이픈 처리되지 않게 된다. 이밖에 특정 단어가 하이픈 처리되지 않게 하려면 그 단어를 \mbox 안에 넣는다.

```
Do not hyphen \mbox{internationalisation}, please. I'm a  
↪ masochistic man.
```

7.5 Tools▷Spell Check(철자 검사)

ℒ_{TEX} 자체는 철자에 대해 알지 못한다. 철자 검사를 위해서는 `ispell`, `aspell` 같은 외부 툴에 의해야 한다.^{*61} UNIX 시스템에서 `ispell`을 다음과 같이 사용할 수 있다.

[역주*61] 한국어 철자 검사는 오픈소스 프로그램의 취약한 부분이다. `hunspell`이 한국어 사전을 제공한다.

```
shell> ispell -t mydocument.tex
```

`-t` 스위치는 `ispell`에게 ℒ_{TEX} 및 ℒ_{TEX} 명령을 무시하라고 요청하는 것이다. 만약 영어가 아니라면 `-d` 스위치로 적절한 언어 사전을 지정해주어야 한다.

```
shell> ispell -d italiano -t mydocument.tex
```

8 Help 메뉴

온라인과 오프라인을 막론하고 ℒ_{TEX} 도움말을 얻는 많은 방법이 있다. 일단 방문해보아야 할 곳은 CTAN 사이트이다. <https://www.ctan.org/tex-archive/info/>.

- UNIX 시스템 명령행에서 `info latex` 명령을 내리면 ℒ_{TEX} 명령과 개념에 대하여 간략하지만 충분한 설명을 제공한다.
- <https://www.ctan.org/tex-archive/info/LatexHelpBook/> ℒ_{TEX}에 대한 매우 훌륭한 도움말로서 Windows에 대한 내용도 잘 되어 있다.
- <https://groups.google.com/group/comp.text.tex/topics> 뉴스그룹을 언급하지 않을 수 없다. 각종 도움을 얻을 수 있는 가치있는 원천이다.

이 문서로 말하자면, 대부분의 GNU/Linux 배포판에서 설치할 수 있는 완전한 ℒ_{TEX}/ℒ_{TEX} 시스템인 TeX Live에 들어 있다. 그밖에 수많은 문서를 제공한다. 나의 Ubuntu 기계의 `/usr/share/doc/texlive-doc/` 디렉터리 아래에서 문서를 찾을 수 있다. 더 나아가, `texdoc` 명령은 굉장히 유용하다. 예를 들어, `fancyvrb` 패키지 문서를 열려면 다음 명령을 내린다.

```
shell> texdoc fancyvrb
```

9 마치는 말

이 문서는 © Guido Gonzato에 의하여 카피레프트로 제공하며 GNU 자유 문서 라이선스로 배포한다. 이 가이드가 도움이 되기를 바란다. 어떤 제안과 코멘트라도 저자에게 보내준다면 매우 기쁠 것이다.

부록 A 문서 템플릿

article 클래스의 템플릿은 2.1절에 보였다. 아래 더 많은 예제를 첨부한다.

```
\documentclass[twoside,11pt]{book}
\begin{document}
\frontmatter
\begin{titlepage}
\title{The Book of Mine}
\end{titlepage}
\author{John B. Smith}
\maketitle
\tableofcontents
\mainmatter
\part{The Beginning}
\chapter{Introduction}
\section{Let's Start}
The book starts here.
\part{The End}
\backmatter
Thank you for reading this book.
\end{document}
```

그림 A.1: Book template

```
\documentclass[twoside,12pt]{report}
% tables and figures at the end:
\usepackage{endfloat}
\begin{document}
\title{Final Report}
\author{John B. Smith}
\date{London, \today}
\maketitle
\begin{abstract}
This is the final report.
\end{abstract}
\tableofcontents
\listoftables
\listoffigures
\part{Start}
\chapter{Begin}
\section{Introduction}
The report starts here.
\end{document}
```

그림 A.2: Report template

```
\documentclass[12pt]{letter}
\begin{document}
\address{My address}
\signature{Guido}
\begin{letter}{John's address}
```

```

\opening{Dear John,}
Thank you for being my friend.
\closing{Hope to see you soon,}
\ps{P.S. Say hello to granny!}
\encl{My son's photographs!}
\end{letter}
\end{document}

```

그림 A.3: Letter template

```

\documentclass[a4paper]{article}
\usepackage{typelcm}
\usepackage{times}
\usepackage{color}
\usepackage{rotating}
\pagestyle{empty}
\begin{document}
\begin{sidewaysfigure}
  \fontsize{2.5cm}{2.5cm}\selectfont
  \centerline{\textcolor{blue}{\textbf{Please:}}}}
  \vskip 1cm
  \fontsize{4cm}{3cm}\selectfont
  \centerline{\textcolor{red}{DO NOT}}
  \centerline{\textcolor{red}{SMOKE}}
  \centerline{\textcolor{red}{HERE!}}
  \vskip 1cm
  \fontsize{2cm}{2cm}\selectfont
  \centerline{\textcolor{magenta}{If you do,}}
  \centerline{\textcolor{magenta}{you'll be \emph{deboned!}}}}
\end{sidewaysfigure}
\end{document}

```

그림 A.4: How to write a notice

```

\documentclass{article}
\usepackage[absolute,showboxes]{textpos}
\usepackage{color}
\usepackage{framed}
\usepackage{graphicx}
\setlength{\TPHorizModule}{10mm} % standard unit of length
\setlength{\TPVertModule}{\TPHorizModule}
\setlength{\TPboxrulesize}{1pt} % box line width
% start everything near the top-left corner
\textblockorigin{0mm}{0mm}
\thispagestyle{empty} % no page number

\begin{document}
\setlength{\parindent}{0pt}
\definecolor{shadecolor}{rgb}{0.9,1,1}
\begin{textblock}{5}(0,0)
% this block is 5 modules wide; height is
% automatically determined

```

```

\begin{center}
  \begin{minipage}[c]{0.8 \linewidth}
    \begin{shaded}
      This block is placed with its top left corner at the 'origin'
      on the page, which has been set to (0mm,0mm). The internal
      margin and the shading are provided by the \texttt{minipage}
      and \texttt{shaded} environments.
    \end{shaded}
  \end{minipage}
\end{center}
\end{textblock}
\begin{textblock}{6}(10,1)
  \includegraphics[width=6cm,angle=-90]{gnuplot.pdf}
  This picture is at (10,1). Note that rotating it
  by -90 makes it overflow the margin.
\end{textblock}
\begin{textblock}{5}[0.5,0.5](2.5,8)
  This block is at position (2.5,8), but because the optional
  argument [0.5,0.5] has been given, it is the centre of the block
  which is located at that point, rather than the top-left corner.
\end{textblock}
\begin{textblock}{3,4}(6,4)
  The dimensions of this block are 3 $\times$ 4 cm.
  Its origin is position (6,4) on the page. Note that the text
  overflows the margin in some cases; you'll want to
  use the \texttt{minipage} environment to prevent that.
\end{textblock}
\end{document}

```

그림 A.5: How to write a poster

부록 B 한글 Markdown과 Pandoc 예제

(이 절은 역자가 추가하였다.)

한글 Markdown 문서를 작성할 때 주의할 점은 KaTeX 패키지를 로드하게 하는 것이다.

```

---
title: 제목
author: 저자
header-includes: |
  \usepackage{kotex}
---

```

만약 XeLaTeX 컴파일이 필요하다면 Pandoc을 실행할 때 다음과 같이 `--pdf-engine` 옵션을 주어야 한다.

```
$ pandoc --pdf-engine=xelatex -s -t latex mywork.md -o mywork.tex
```

2.4절에서 소개하고 있는 Pandoc 변환기를 이용하는 Markdown 예제를 한글로 작성하는 샘플을 제공한다. 해당 절의 샘플을 옮긴 것이다.

```

---
title: |
    제목을 여기에 적는다: \
    제목을 두 줄 이상 적을 때
author:
    - Guido Gonzato
    - KTUG (번역)
date: 2018년 1월 (2025년 10월)
documentclass: oblivoir
papersize: b5
fontsize: 11pt
classoption: gremph
numbersections: true
header-includes: |
    \usepackage{fapapersize}
    \usefapapersize{*,*,30mm,50mm,25mm,*}
    \ifLuaOrXeTeX
    \setkomainfont(Noto Serif CJK KR)(* Bold)(Noto Sans CJK KR Light)
    \fi
abstract: |
    요약문을 여기에 적어보자.
---

<!-- 코멘트는 이런 방법으로 적어넣음 --->

# 섹션 헤더

글자를 볼드 처리할 수 있다.
*기울임(italic)* 처리도 가능한데 한글이 이를
지원하도록 설정해야 한다. 여기서는 고딕체를 찍도록
설정하였다. 일반 텍스트는 그냥 글을 써넣으면 되겠다.
그만 읽고 싶으면 [마지막 섹션](#theend)으로 이동해보아라.

## 서브섹션 헤더

`verbatim text`도 써넣을 수 있다.

# 마지막 섹션 {#theend}

아버지께서 말씀하시기를:

> 이런, pandoc이 아직 발명되지 않았다니!

그렇지만 지금은 pandoc이 있지.

<!-- end of pandoc_template_ko.md -->

```

문서 클래스를 oblivoir로 하였다. 보통은 article 문서가 된다. header-includes
 부분에서 oblivoir를 위한 설정이 되어 있으므로 다른 문서 클래스라면 이 부분을 해당 클래
 스에 맞도록 수정하거나 제외한다.

Pandoc으로 pdf 변환하려면 다음 명령을 내리는데,

```
shell> pandoc --toc --pdf-engine=xelatex \  
pandoc_template_ko.md -o pandoc_template_ko.pdf
```

--pdf-engine을 xelatex으로 지정해주었다. 이 스위치가 없으면 pdflatex으로 컴파일한다.

YAML header의 numbersections: 는 섹션에 번호를 붙이라는 것이다. 명령행 옵션으로 --number-sections 인자를 주어도 같은 결과를 얻는다. Pandoc의 기본값은 섹션 번호를 억제하는 것이다.

만약 TOC가 필요하다면 Pandoc 실행 명령 옵션으로 --toc하면 충분하다.