워드 프로세서 사용자를 위한 LaTeX
version 1.0.7

Guido Gonzato, Ph.D.
guido.gonzato@univr.it
Università di Verona (Italy)
Direzione Informatica

김강수 (옮김)

2010년 9월 14일 2010년 9월 18일

요 약

LaTeX 텍스트 처리는 워드 프로세서를 사용하는 것에 비해 상당한 장점이 있다. 그러나 초보자들은 그 일을 어떻게 해야 하는지, 필요한 특정 기능을 어디서 찾아야 하는지 알아내기가 어렵다. 이 안내서는 워드 프로세서와 LaTeX 조판을 비교함으로써 워드 프로세서에서 LaTeX으로 이행하는 것을 도와주려 한다. 주요 워드 프로세서 기능을 열거하고 각각이 LaTeX에서 어떻게 실현되는가를 보여준다. 많은 예를 첨부하였다.

역자는 이 문서에 두 가지로 개입하였다. 본문을 충실히 번역하는 이외에 역자의 의견을 방주 형태로 추가하였다. 그리고 일부 소절이나 단락을 추가한 것도 있다. 원칙적으로 원문은 손상하지 않았으며 교정하거나 코멘트할 것이 있으면 모두 위와 같은 형식에 의해서 했다. 본문에 추가한 단락과 소절은 그 사실을 방주에서 밝혔다.

차 례

표 차례

그림 차례

1 서론

시작하기 전에 이 안내서가 LaTeX 입문서가 **아니라**는 점을 말해두고자 한다. 이 문서를 읽고 있다는 것은 적어도 LaTeX이 무엇인지, 기본 명령은 어떤 것이 있는지를 알고 있다는 뜻이다. 이 안내서에서 내가 설명하려 하는 것은 LaTeX을 사용함으로써 어떻게 워드 프로세서를 효과적으로 대체할 수 있는가 하는 것이다.

워드 프로세서는 오늘날 사무자동화에서 '킬러 프로그램'이다. 익숙한 WYSIWYG 인터페이스를 가지고 있기 때문에 LaTeX에 비해 더 쉬운 것으로 여겨진다. 평균적인 사무보조원은 꽤 짧은 시간에 그 사용법을 익힌다. 문제는 이 물건이 날이 갈수록 느려지고 비대해지고[1] 버그투성이가 되고 심심하면 죽고 비싸고 바이러스에 감염되고 서로간에 호환불가능하게 되어 간다는 것이다. 그 기본 출력 품질에 대해서는 말하지 않겠다.

LaTeX은 훌륭한 대인이다. 그렇지만 당신이 작성하려 하는 것이 즉흥적이고 비구조적인 문서라면 무슨 방법이 있는 걸까.

요약하면 이따금 워드 프로세서와 같은 기능을 쓰고 싶은데 그걸 LaTeX으로 하고 싶을 때가 있다. 한때 좋아했던 :-) 워드 프로세서로 했던 작업을 LaTeX으로 어떻게 하면 되는지 알 수 있다면 좋을 것이다.

이것이 내가 이 짧은 안내서를 쓰게 된 동기이다. 이미 말한대로 나는 LaTeX의 기본을 알고 있는 사람을 전제로 하고 있다. 만약 당신이 진정한 초심자라면 http://www.ctan.org/starter.html로 가서 'The (Not So) Short Introduction to LaTeX 2ε'를 읽어볼 것을 권한다. 그 외의 좋은 처음 시작 문서로 http://en.wikibooks.org/wiki/LaTeX/이 있다.

한국어 번역본 〈그다지 짧지 않은 LaTeX 2ε〉를 읽으시라.

이어지는 절에서 어떤 가상의 워드 프로세서를 상정하고 그 메뉴와 메뉴 항목을 살펴보면서 각각에 대응하는 LaTeX 방식을 찾아보겠다. 자신이 LaTeX 순수주의자라서 이런 식의 접근이 지겹게 느껴진다면…… 맘상하지 말고 여기서 접어라.

1.1 기본사항

워드 프로세서의 기능 가운데 많은 부분은 에디터의 몫이다. 그밖에 LaTeX 명령에 의해 구현되는 것이 있고 패키지(*packages*)에 의해 이루어지는 것이 있다. 패키지란 LaTeX을 확장하여 새로운 명령과 환경을 제공하는 매크로 모음이다. 수많은 패키지가 존재하는데 유일한 문제는 그게 어디에 있는지, 무슨 일을 하는지, 그리고 사용하려면 어떤 절차를 거쳐야 하는지를 알아야 한다는 것이다. 패키지에 대해서는 1.1절에서 더 다룬다.

\usepackage 명령으로 쓸 수 있는 것을 보통 '패키지'라 한다. 이것은 스타일 파일(.sty)과 설정 파일 및 사용법 문서로 이루어져 있다.

패키지를 포함하여 그밖의 TeX에 관련된 자료들은 CTAN(the Comprehensive TeX Archive Network)을 구성하는 여러 사이트를 통해 이용할 수 있다. http://www.ctan.org 사이트를 이미 소개했다. 이 사이트는 여러 곳의 미러 사이트를 가지고 있다. 이제부터 CTAN:이라 하면 "자신이 선호하는 CTAN 미러 사이트의 TeX 디렉터리"를 가리키는 것으로 하겠다. 예를 들면 자신의 플랫폼에 적합한 LaTeX을 얻기 위해서는 CTAN://systems (여기서는 http://www.tex.ac.uk/tex-archive/systems/)를 접속하면 된다.

KTUG에서도 CTAN을 미러링한다.

문서 작성을 위해서는 좋은 텍스트 에디터가 필요하다. 초심자에게 더 좋은 것은 *LaTeX*

[1] 옛날에 나는 나의 학위논문을 128k 램을 가진 Z80 가정용 컴퓨터에서 작성했다. WordStar 워드 프로세서와 나의 논문이 단면 CP/M 부팅가능 720k 플로피 디스크 한 장에 다 들어가고도 남는 공간이 많았다.

shell, 즉 소스를 작성하고 미리보기를 할 수 있는 등 LaTeX에 필요한 기능을 갖춘 에디터이다.

아래에 나열된 프로그램은 추천할 만한 것이다. 이 모두가 Free/Open Source 소프트웨어들이다.

- Texmaker (multiplatform):
  http://www.xm1math.net/texmaker/index.html
- TeXworks (multiplatform):
  http://tug.org/texworks/
- LyX, an almost-WYSIWYG LaTeX editor (multiplatform):
  http://www.lyx.org/
- TeXShop (Mac OS X):
  http://www.uoregon.edu/~koch/texshop/
- TeXnicCenter (Windows):
  http://www.texniccenter.org/

Windows를 위한 LaTeX 용 편집기의 완전한 목록은 http://home.arcor.de/itsfd/texwin에서 찾을 수 있다. 매킨토시 상의 LaTeX에 관한 정보를 얻으려면 http://www.esm.psu.edu/mac-tex/을 참고하라.

에디터가 지원하는 기능

LaTeX은 조판기일 뿐이다. 잘라붙이기, 찾기와 바꾸기 등은 에디터에게 맡겨진다. 표 1은 geek들에게 유명한 편집기인 GNU `emacs`와 `vim`의 기본 키 바인딩과 Borland IDE 키 바인딩을 적용한 `jed`의 주요 명령을 요약한 것이다.

패키지 추가

아래의 사항은 `TeX Live`에 적용된다. `TeX Live`는 대부분의 GNU/Linux 배포판에 포함되어 있다. MacTeX에도 적용될 것이지만 내가 직접 경험해보지는 못했다. MiKTeX(아마 가장 유명한 Windows 텍 시스템)을 위한 안내는 그 뒤에 이어진다.

방대한 분량의 LaTeX 패키지들이 기본으로 지원된다. 예를 들면 Ubuntu는 많은 `texlive-*` 패키지를 제공한다. 만약 지원되지 않는 패키지를 사용해야 한다면 아래와 같이 하라.

Ubuntu는 한글 LaTeX 패키지 ko.TeX도 지원한다. 대부분의 필요한 패키지는 `TeX Live` 자체가 지원하는 것으로 충분하다. Ubuntu 이외의 Windows나 MacTeX TeX Live에서는 패키지의 설치와 삭제를 위하여 `tlmgr`을 사용한다.

Windows용 TeX Live의 경우는 `%USERPROFILE%\texmf` 아래에 본문의 설명과 같은 티렉터리 구조를 만든다. 이 폴더 아래 있는 파일들은 `mktexlsr`을 실행하지 않아도 된다. 단, Windows라도 `%HOME%` 변수가 설정되어 있다면 그 아래로 가야 한다.

1. 다음과 같은 디렉터리 구조를 만든다.

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

   이 디렉터리 아래 새로운 패키지를 인스톨한다.

2. 가까운 CTAN 미러 사이트에서 패키지를 (대부분 zip-압축 디렉터리이다) 내려받는다. 예컨대 `foo.zip`이라 하자.

Action	**Emacs**	**Vim**	**Jed**
command mode	Alt -X	Esc	Alt -X
insert mode	n/a	i a o O	n/a
line editor mode	n/a	:	n/a
	파일 조작 *file operations*		
open file	Ctrl -X Ctrl -F	:e	Ctrl -KE
insert file	Ctrl -Xi	:r	Ctrl -KR
save file	Ctrl -X Ctrl -S	:w	Ctrl -KD
save as	Ctrl -X Ctrl -W name	:w name	Ctrl -KS
close file	Ctrl -XK	:q	Ctrl -KQ
change buffer	Ctrl -XB	bN	Ctrl -KN
undo	Ctrl -XU	u	Ctrl -U
redo	Ctrl ^	Ctrl -R	Ctrl -G Ctrl -U
exit	Ctrl -X Ctrl -C	:qa!	Ctrl -KX
	이동 *moving around*		
word left	Alt -B	b	Ctrl -A
word right	Alt -F	w	Ctrl -F
start of line	Ctrl -A	0	Ctrl -QS
end of line	Ctrl -E	$	Ctrl -QD
page up	Alt -V	Ctrl -U	Ctrl -R
page down	Ctrl -V	Ctrl -D	Ctrl -C
start of buffer	Alt -<	1G	Ctrl -QR
end of buffer	Alt ->	G	Ctrl -QC
line n.	Alt -G n.	n.G	Ctrl -QI
	삭제 *deleting*		
character left	Ctrl -H	X	BS
character right	Ctrl -D	x	Alt -G
word left	Alt -DEL	db	Alt -BS
word right	Alt -D	dw	Ctrl -T
end of line	Ctrl -K	d$	Ctrl -QY
line	Ctrl -A Ctrl -K	dd	Ctrl -Y
	찾기와 바꾸기 *search & replace*		
search	Ctrl -S text	/text	Ctrl -QS
replace	Alt -%	:s/old/new/g	Ctrl -QA
	선택영역 *blocks*		
start selection	Ctrl -	v	Ctrl -KB
cut	Ctrl -W	D	Ctrl -KY
copy	Alt -W	Y	Ctrl -KH
paste	Ctrl -Y	P	Ctrl -KC

표 1: Emacs, Vim, Jed의 유용한 단축키

3. 적당한 곳에 압축을 푼다.

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

4. 만약 `.sty` 파일이 없으면 `latex foo.ins` 또는 `latex foo.dtx` 를 실행하여 만들어내게 한다.

5. `texhash ~/texmf` 명령을 실행한다.

이 지침은 MiKTeX 2.4 이전 버전에 해당하는 것 같다. 현재 버전에서는 위치가 조금 다르다. 사용자 패키지를 MiKTeX 2.8 또는 2.9에서 설치하기 위해서는 두 가지 방법이 있는데 하나는 %APPDATA% 또는 %PROGRAMDATA% 아래 설치된 사용자용 texmf 트리를 이용하는 것이고 다른 하나는 이전과 비슷하게 localtexmf를 추가하는 것이다. 다만 두번째 경우 반드시 사용자가 TeX roots를 추가하는 조작을 해주어야 한다.

MiKTEX에 새로운 패키지를 추가하려면 `\latex\newpackage`를 `C:\localtexmf\tex\` 아래 만들고 거기에 필요한 파일을 가져다 둔다. 앞서와 같은 과정을 진행한 다음 MiKTeX Options를 실행하여 Refresh now 버튼을 클릭한다. 또는 `initexmf -u` 명령을 실행한다. 그것으로 되었다.

일단 패키지가 설치되면 `documentclass` 선언 이후에 다음 한 줄 추가하는 것으로 자신의 문서에 사용하게 할 수 있다.

```
\usepackage{foo}
```

1.2 명심할 사항

시작하기 전에, 다음 사항을 꼭 기억하자.

1. 문서의 **구조화**에 익숙해져야 한다. part, chapter, section와 같은 구조적 요소에 신경을 쓰자. 학술 문서를 작성하는 경우가 아니라도 이것은 중요하다.

2. LATEX은 확실히 포매팅 파라미터로 문서를 난잡하게 만드는 것을 기피하게 한다. **모양**에 너무 신경쓰지 말고 **내용**에 집중하라.

스스로 확신을 가지고 위의 황금률을 적용해보라. 그러면 인쇄된 문서가 신기하게도 전문가가 만든 것처럼 보일 것이다. 다시 강조하거니와 진정한 LATEX 초심자가 되어 공부해볼 것을 권장한다.

이 규칙은 대부분의 문서에 적용된다. 그러나 주로 비구조적인 문서(회람, 쪽지 등)를 작성해야 하고 그 방법을 꼭 찾고 싶다면, 이 안내서를 계속 읽어보기로 하자.

2 File 파일 메뉴

이 메뉴에 속하는 항목 중 많은 것이 자명하게 LATEX과는 별 상관이 없다. File/Open, File/Save, File/Close 등은 에디터에서 해줄 일들이다.

2.1 File/New 파일/새 문서

빈 문서에 해당하는 LaTeX 소스는 다음과 같다.

```
\documentclass{article}
\begin{document}
% This is a comment. Write your stuff here.
% 이것은 주석문입니다. 내용을 쓰세요.
\end{document}
```

LaTeX으로 쓰여진 문서는 본질적으로 구조화되어 있다. 좀더 현실적인 보기는 다음과 같다.

```
\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.
\end{thebibliography}
\end{document}
```

더 많은 문서 표본이 부록 A에 있다.

한글 문서 표본도 수록해 두었다.

2.2 File/Save As··· 파일/다른 이름으로 저장

다음 도구들은 LaTeX을 다른 포맷으로 변환할 때 유용하다.

- TeX4ht은 아마도 가장 훌륭한 LaTeX–HTML 변환기일 것이다.
  http://www.cse.ohio-state.edu/~gurari/TeX4ht/
- latex2html, 또다른 HTML 변환기이다.
  http://saftsack.fs.uni-bayreuth.de/~latex2ht/,
  CTAN://support/latex2html
- latex2rtf, Rich Text Format으로 변환기이다.
  CTAN://support/latex2rtf

- detex은 명령행 변환기인데 모든 LaTeX 태그를 제거하여 플레인 텍스트로 만들어 준다.
  http://www.cs.purdue.edu/homes/trinkle/detex/,
  CTAN://support/detex/

그밖에 PDF 변환에 관한 자세한 사항은 2.7절을 참고하라.

2.3 File/Save As Template 파일/본보기문서로 저장

LaTeX '본보기 문서'로 저장한다는 것은, 아마도 새로운 LaTeX 패키지를 만든다는 것과 같은 말일지도 모른다. 만약 그렇다면 그것은 너무 복잡한 문제라서 이 가이드의 범위를 넘어선다.

2.4 File/Import 파일/불러오기

다음 도구들은 다른 포맷으로부터 LaTeX으로 변환한다.

sourceforge에 rtf2latex2e라는 것이 있는데 사용 언어에 제한이 있고 한글은 포함되어 있지 않다.

- rtf2latex: CTAN://support/rtf2latex
- html2latex: CTAN://support/html2latex
- wvware는 MS Word를 LaTeX을 포함하여 여러 포맷으로 변환하는 도구의 모음이다. http://wvware.sourceforge.net
- Abiword는 프리 워드 프로세서이다. http://www.abisource.com, MS Word 문서를 불러올 수 있고 LaTeX으로 저장 가능하다.
- txt2tex: CTAN://support/txt2tex 플레인 텍스트 파일을 LaTeX으로 변환하는데 꽤 좋은 결과를 보여준다.

그밖의 *2latex 컨버터를 같은 주소에서 찾아볼 수 있다.

또 한 가지 재미있는 것은 OOoLatex이라는 OpenOffice 확장 매크로이다. http://ooolatex.sourceforge.net

OpenOffice 확장과 관련해서 writer2latex 확장모듈이 있다. http://writer2latex.sourceforge.net

2.5 File/Page Setup 파일/페이지 설정

페이지 크기, 방향, 여백을 설정하는 일반적인 방법은 \documentclass의 인자로 이를 지정하는 것이다. 페이지 크기는 a4paper, a5paper, b5paper, letterpaper, legalpaper, executivepaper 중에서 고를 수 있고, 방향은 portrait가 디폴트이며 landscape를 사용할 수 있다. 예를 들어보자.

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

문서 전체에 걸친 여백 설정은 다음 명령으로 한다.

```
\setlength{\leftmargin}{2cm}
\setlength{\rightmargin}{2cm}
\setlength{\oddsidemargin}{2cm}
\setlength{\evensidemargin}{2cm}
\setlength{\topmargin}{-1cm}
\setlength{\textwidth}{18cm}
\setlength{\textheight}{25cm}
```

geometry 패키지는 paper size, margin width 등의 파라미터를 완전히 제어하게 해준다. geometry는 여기에 모두 열거하기에 너무 많은 옵션이 있으므로 패키지 문서를 꼭 읽어보기 바란다. 아래 보인 예는 간단한 사용법으로서 이 중 몇 가지 파라미터는 상충하는 것도 있으며 보기를 보일 목적으로 지정된 것일 뿐이다.

memoir에서는 geometry를 이용하기 보다 memoir 자체의 레이아웃 설정 기능을 이용하는 것이 좋다. oblivoir에는 fapapersize라는 간편한 판면 설정 유틸리티가 제공된다.

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

선택사항은 다음과 같이 지시할 수도 있다.

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

Page Setup/Headers and Footers 페이지 설정/머리말과 꼬리말

fancyhdr 패키지는 \pagestyle{fancy}라는 새로운 명령을 제공한다. 이것은 현재 섹션(또는 book.cls에서는 챕터)과 서브섹션으로 헤더를 만들고 페이지 번호를 바닥에 찍어준다. 제법 팬시하다. 머리말과 꼬리말은 당연히 사용자가 변경할 수 있다. 세 부분으로 이루어지는데, 각각 왼쪽으로 정렬되는 부분, 가운데 오는 부분, 오른쪽으로 정렬되는 부분이다. 이들을 사용자가 설정하려면 다음 보기와 같이 한다.

memoir 또는 oblivoir 클래스는 fancyhdr와는 다른 방식으로 페이지 스타일을 설정한다. 훨씬 직관적이고 강력한 커스터마이징이 가능하다. 자세한 것은 memoir 사용설명서를 참고하라. 여기에 주석을 붙이는 이유는 fancyhdr와 일부 상충하는 경우가 있으므로 이를 미리 알려두고자 함이다. 즉, memoir에서는 fancyhdr를 쓰지 않는 것이 좋다.

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

2.6 File/Printer Setup 파일/프린터 설정

이것은 운영체제 의존적인 문제로서 LaTeX과는 아무 상관없는 것이다. 만약 Unix 계열의 시스템을 사용한다면 다음 팁이 도움이 될 것이다.

- `lpr -P printername` 특정 트린터로 프린트한다.
- `lpr -# 10` 10장을 인쇄한다.
- `lpr -r` 인쇄 후에 파일을 삭제한다.

더 많은 팁에 대해서 아래를 볼 것.

2.7 File/Print Preview 파일/인쇄 미리보기

LaTeX 입력 파일이 준비되었다면 다음 중에서 선택할 수 있다.

- `.dvi`로 변환(`latex file.tex`)하여 `xdvi`나 `yap`과 같은 프리뷰어로 미리보기할 수 있다.
- `.dvi`를 `dvips`를 통하여 PostScript로 변환한다. 그런 다음에 `Ghostview`와 같은 프로그램으로 미리보기한다.
- `dvipdf`를 이용하여 `.dvi`를 `.pdf`로 변환하거나 직접 `pdflatex`으로 `.pdf` 파일을 생성한다.

실제 `.dvi`나 `.ps` 파일 출력은 현재 거의 문제삼지 않게 되어 간다. 예를 들어 차세대 TeX 엔진이라 하는 LuaTeX이나 XeTeX은 `.pdf` 출력이 기본이다.

내 생각에, `.pdf` 파일을 만드는 것이 가장 좋다. 활용가능성이 가장 폭넓기 때문이다.

`dvipdf`가 `.dvi`를 `.ps`를 거쳐 `.pdf`로 만드는 스크립트인 데 반해, `pdflatex`을 사용하는 것은 더 흥미롭다. 사실 hyperref이나 url과 같은 패키지들은 `.pdf` 파일이 브로우징 가능하게 만들어준다. 4.15절을 보라. 그러나 `pdflatex`을 사용하려면 주의해야 할 점이 좀 있다. 다른 패키지와의 호환성 문제를 경험할 수도 있기 때문이다. 자세한 것은 4.12절을 보라.

만약 타겟 출력이 `.pdf`가 되면 할 수 있는 일이 무척 많다. 그 가운데 하나로 annotation이나 layer를 직접 넣어서 풍부한 `.pdf`를 만드는 것이 가능하다. annotation은 pdfcomment 패키지를 이용하여 손쉽게 작성이 가능하다. 한편, 일부 pdf의 특별한 기능을 겨냥한 패키지들이 PDFLaTeX에서만 동작하는 것이 많았으나 최근 XeTeX과 LuaTeX에서도 같은 결과를 얻을 수 있도록 업그레이드되어 가고 있다. 멀티미디어를 pdf에 내장하는 media15와 같은 패키지가 그 한 예이다.

2.8 File/Print 파일/인쇄

(Unix 계열 운영체제에서) 간단히 `lpr file.ps`를 명령행에서 주거나 프리뷰어의 File/Print 메뉴 항목을 선택하면 된다.

2.9 File/Versions 파일/버전관리

version 패키지가 LaTeX 소스의 버전관리를 위한 기본적인 기능을 제공하기는 하지만 RCS (Revision Control System)이나 CVS (Concurrent Version Control System)와 같은 외부 프로그램을 사용하는 것이 낫다. 에디터와 연동할 수 있으면 더 좋다. CVS와 RCS에 대한 간단한 소개로 http://www.faqs.org/docs/Linux-HOWTO/CVS-RCS-HOWTO.html을 찾아보기 바란다.

최근 SVN이 주목받고 있다.

3 Edit 편집 메뉴

이 메뉴는 LaTeX 기능보다는 에디터와 더 많이 관계된다. **편집/잘라내기**, **편집/복사하기**, **편집/붙이기**, **편집/찾기**, **편집/바꾸기**와 같은 에디터에 공통되는 항목들에 대한 단축키는 표 1에서 이미 보였다.

텍스트 일부를 선택하는 것은 자르기, 붙이기를 위해서이기도 하지만 선택된 텍스트에 특정 스타일을 적용하기 위해서이기도 하다. 이에 대응하는 LaTeX의 작용은 텍스트 일부를 중괄호나 환경(environment)으로 감싸는 것이다. 예를 들면 텍스트 일부에 두꺼운 글씨 속성을 부과하려면 다음 가운데 한 가지 방식을 쓰면 된다.

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

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

3.1 Edit/Autotext 편집/자동 완성

자동 완성이란 예를 들어 'PS'라고 입력하면 'PostScript'라고 자동으로 입력되는 기능을 말한다. 이것도 에디터의 역할이지만 대략 여기에 해당하는 LaTeX 기능이 있다.

```
\def\PS    {\textsc{PostScript}}
```

이렇게 하면 \PS라고 입력하는 곳마다 \textsc{PostScript}에 해당하는 PostScript가 찍힌다. 대소문자 구별에 주의하자.

일반적으로 \def 보다 \newcommand를 쓰는 것이 더 안전하다.

4 Insert 삽입 메뉴

4.1 Insert/Breaks 삽입/나누기

- 행이 잘라지지 않는 강제 공백 한 칸은 ~(tilde)로 나타낸다.
- 행나눔을 강제하려면 \linebreak나 \newline을 쓴다. 이 둘의 차이는 아래를 보라. \\도 새 줄을 시작한다. \\[1cm]와 같이 길이를 지시하면 문단 사이의 간격을 조절할 수도 있다.
- 새 문단은 빈 줄 하나를 넣고 시작한다. \par 명령과 같다.
- 끝으로 강제 페이지 나누기는 \newpage나 \clearpage를 쓴다.

원문에는 \\를 새 문단으로 소개하고 있는데, 이것은 \par와 같은 것이 아니라 \newline과 같은 것이다. 설명하는 순서를 조금 바꾸었다.

\linebreak와 \newline의 차이는, 앞의 것이 행의 나머지를 다 채우고 다음 행을 시작한다는 것이다. 다음을 보자.

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

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

또한, \clearpage는 \newpage와 마찬가지로 새 페이지를 시작하지만 그 시점까지 출력되지 않고 대기중인 *floats*들, 즉 figure나 table들을 모두 출력한 다음에 새 페이지를 만든다는 점이 다르다. float에 대해서는 4.12절에서 설명한다.

4.2 Insert/Enumerated List 삽입/문단머리표

숫자나 기호 붙은 리스트는 itemize와 enumerate 환경에 해당한다. 리스트 환경에서 글머리에 붙는 기호를 바꾸려면 \item 명령의 인자로 특정하면 된다.

* with an asterisk;
- with a dash;
. with a dot.

```
\begin{itemize}
  \item[*] with an asterisk;
  \item[-] with a dash;
  \item[.] with a dot.
\end{itemize}
```

다른 방법은 첫째 수준에서 넷째 수준까지 각 수준의 숫자에 대응하는 카운터 스타일을 재정의하는 것이다.[2] 숫자 표현에 몇 가지 스타일이 있는데, \arabic은 '보통' 숫자이고 \roman은 로마 숫자 소문자(예를 들면 8을 viii로 표시), \Roman은 로마 숫자 대문자, \alph와 \Alph는 각각 알파벳 문자 소문자와 대문자이다. 숫자 형식 가운데 \fnsymbol에 대해서는 나중에 설명한다.

* first level, item 1
* first level, item 2
    - second level, item 1
    - second level, item 2
* first level, item 3

```
\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}
```

말하자면, 숫자붙인 문단의 숫자를 로마 숫자와 알파벳 대문자로 하려면 다음과 같이 한다.

A first level, item 1
B first level, item 2
    i second level, item 1
    ii second level, item 2
C first level, item 3

```
\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}
```

또다른 방법으로 enumerate 패키지를 사용할 수 있다. 이 패키지는 enumerate 환경을 재정의하여 옵션 인자로 주어진 형식을 쓰게 한다. A a I i 1 가운데 한 글자가 오면 그것을 카운터 값에 각각 \Alph, \alph, \Roman, \roman, \arabic이 주어진 것으로 대치한다.

[2] 각각 \labelitemi, \labelitemii, \labelitemiii, \labelitemiv라는 매크로 이름을 갖는다.

그밖의 글자로 이루어진 텍스트를 글머리에 쓰려면 중괄호로 해당 텍스트를 묶어주면 된다.

example I. First example.

example II. Second example.

example III. Last example. Go to Item I.

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

문단 속에 포함된 리스트를 쓰려면 paralist 패키지를 이용한다. 이 패키지도 enumerate와 마찬가지로 문단 머리를 선택 인자로 지정할 수 있다. inparaenum 환경을 쓸 수 있다.

ko.TEX에는 enumerate에 해당하는 dhucs-enumerate, paralist에 해당하는 dhucs-paralist 패키지를 제공하고 있으며 이 패키지들은 한글식의 문단머리를 지정할 수 있게 하고 있다. 또한 oblivoir를 위해서 xob-paralist가 별도로 있고 dhucs-enumerate는 oblivoir에 기본적으로 포함되어 있다. memoir가 enumerate 패키지를 이미 포함하고 있는 것과 같다. 여기에 추가된 글머리표지는 가, ①, (1), (a), ⓐ, ⅰ, Ⅰ, ㄱ, ㉠, ㉮, (ㄱ), (㉮)의 열두 개이다.

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.

```
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}
```

이상에서 본 대로, `A a I i 1` 문자는 카운터를 수식하기 위해 사용된다. paralist는 더 많은 일을 할 수 있다. 해당 패키지 문서를 읽어볼 것을 권장한다.

한글 식 paralist가 어떻게 구현되는지를 여기서 잠깐 보고 가기로 하자. 이것은 xob-paralist 패키지로 한 것이다.

이 문단은 역자가 추가하였다.

문단 안에서 항목을 나열하고자 한다. ㄱ) 사과, ㄴ) 배, 그리고 ㄷ) 복숭아. 다른 방식의 라벨을 달아본다. ① 사과, ② 배, 그리고 ③ 복숭아. 첫번째 아이템을 참조하면, ㄱ.

```
문단 안에서 항목을 나열하고자 한다.
\begin{inparaenum}[ㄱ)]
  \item 사과, \label{firstk}
  \item 배, 그리고
  \item 복숭아.
\end{inparaenum}
다른 방식의 라벨을 달아본다.
\begin{inparaenum}[①]
  \item 사과,
  \item 배, 그리고
 \item 복숭아. 첫번째 아이템을 참조하면, \ref{firstk}.
\end{inparaenum}
```

4.3 Insert/Special Character 삽입/특수문자

이 한 문단은 역자가 상세히 설명하기 위해 부연하였다.

먼저 LaTeX 입력 파일을 만들 때 몇 가지 글자는 입력 방법이 정해져 있다는 것을 상기하자. 예를 들면 여는 따옴표는 ``로, 닫는 따옴표는 ''로 입력하는 것이 관행이다. 그리고 --는 en-dash(–), ---는 em-dash(—)를 입력하는 방법이다. '{'과 '}'는 특히 중요한데 이것은 LaTeX에서 '범위'를 설정하는 기호로 쓰이기 때문에 이대로 입력하여도 텍스트에

Character	LaTeX Sequence
$	\$ 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: 몇 가지 특수문자를 입력하는 방법

나타나지 않는다. 이 괄호 기호를 텍스트로 나타내고 싶다면 반드시 \{와 같이 입력해야 한다. LaTeX의 문법과 관련된 특수기호(이른바 '예약문자')는 { } ^ _ # $ % ~ \ &와 같은 것이 있다. 이 문자를 그대로 입력하면 오류를 만날 수 있다.

이와 같이, 몇 가지 문자들은 LaTeX에서 특별한 의미를 갖는다는 것을 명심해야 한다. 이 글자들을 문장에 나오게 하려면 \를 앞에 붙이거나 수학 모드에서 쓰거나 하는 등 특별한 방법으로 입력하여야 한다. 표 2를 보라. 이 표에 있는 일부 명령은 textcomp 패키지가 필요하다.

XeTeX에서는 xltxtra 패키지를 textcomp 대신 쓸 수 있다. textcomp 패키지에 관한 언급은 역자의 추가임.

유럽어의 액선트 붙은 문자를 예를 들어 \'e와 같이 입력하여 'é'를 얻는 것은 비슷하지만 이와는 또다른 문제이다. 이에 대해서는 7.3절을 보라.

ASCII 코드가 아니라 유니코드를 사용하는 XeTeX과 같은 엔진에서는 이 방법이 뜻대로 되지 않을 수 있다. 대체로 위험하므로 이런 방법은 되도록 쓰지 않는 것이 좋다.

특수 문자를 입력하는 또 한 가지 방법은 해당 ASCII 코드를 \char 명령으로 주는 것이다. 예를 들면 $ & ^ ~을 얻으려면 \char36 \char38 \char94 \char126이라고 입력한다.

특별한 글자와 심볼을 많이 제공하는 패키지가 있다. 예를 들면 pifont는 \ding, \dingfill, \dingline, \dinglist와 같은 명령을 제공한다. 첫번째의 \ding 명령은 특정 코드에 해당하는 딩뱃 문자를 찍어준다. 다른 명령들은 각각 \fill, \line, \list 명령과 환경에 해당하는 것으로 특정 딩뱃 코드를 인자로 주도록 되어 있다.

☞ one

☞ two

☞ three

```
\begin{dinglist}{43}
  \item one
  \item two
  \item three
\end{dinglist}
```

다음 보기는 좀더 그럴싸하다.

① one

② two

③ three

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

심볼 문자는 여기서 언급하기에 너무 그 수가 많아서 차라리 'The Comprehensive LaTeX Symbol List'라는 문서를 읽어보는 편이 낫다. CTAN://info/symbols/comprehensive 에서 찾을 수 있다.

대부분의 TeX 배포판에서 texdoc을 이용하여 texdoc symbols라고 명령행에서 치면 바로 읽을 수 있다.

유로화 기호(€)

공식 유로화 기호는 eurosym 패키지가 제공한다. 이것은 다음 두 가지 방법으로 사용 가능하다.

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

XeTeX에서는 굳이 eurosym이나 marvosym을 쓸 것 없이 바로 유로화 기호를 €([U+20AC]) 입력하거나 \char"20AC와 같이 입력할 수도 있다. 일반적으로 xltxtra가 제공하는 \texteuro(€)를 쓰는 것이 좋다. 요즘 대부분의 폰트에는 유로화 기호가 들어 있다. TnXTeX에서는 marvosym을 쓸 수 없으므로 이런 식으로 유로화 기호를 바로 얻도록 소스를 일부 수정하였다. 그 결과 이 절에서 설명하는 유로화 기호의 모양의 변화가 출력물에 반영되지 않을 수 있다.

둘 다 \euro 명령을 제공하며 결과는 €로 나타난다. 이 기호가 실제 찍히는 모양은 [gen] 옵션을 주느냐 [official] 옵션을 주느냐에 달려 있는데, [gen]의 경우는 €로 찍히고 [official]의 경우는 €로 찍힌다. 차이점을 눈여겨 보라. 두번째 것은 \officialeuro 명령으로도 얻을 수 있다.

marvosym 패키지도 유로 기호를 제공한다. 이 패키지는 이외에도 꽤 많은 멋진 기호 문자를 포함하고 있다. 이 패키지를 쓸 때 유로화 기호는 \EUR 명령으로 € 과 같은 결과를 얻는다.

4.4 Insert/Formula 삽입/수식

LaTeX은 특히 수식 조판에 강하다. 수학 기호를 텍스트 속에 넣으려면 그것을 $로 감싸주어야 한다.

$는 시작기호와 끝기호가 같아서 흔히 실수하게 하는 원인이 된다. 이를 피하고 싶으면 문장중 수식을 \(와 \)로 감싸주는 편이 낫다.

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

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

displaymath와 equation은 별행 수식을 식자한다. 뒤의 것은 나중에 참조할 수 있도록 수식 번호를 붙여준다.

displaymath는 \[와 \]로 줄여쓸 수 있다. displaymath나 equation은 그 안에서 줄나누기가 안 되는데 이것은 $\mathcal{AMS}$-math를 이용하여 손쉽게 구현할 수 있다. 수식 입력에 관한 사항은 별도의 참고문헌을 보아야 한다. 특히 mathmode라는 문서가 유명하다.

Fermat's Last Theorem is defined as:

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

Can you prove Eq. 1?

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

수학기호 중에 ~이 있다. 이것은 \sim으로 얻는데 그 모양이 우리글의 문장부호인 ~(물결표)와 비슷하게 생겨서 흔히 대용하는 경우가 많으나, 엄연히 다른 부호이므로 넘나들어 써서는 안 된다.

4.5 Insert/Footnote 삽입/각주

`\footnote[n]{footnotetext}` 명령으로 충분하다. 선택적 인자인 `[n]`은 각주 번호를 수정한다. `\footnote` 명령은 단어 뒤에 이어붙는 반점, 온점, 그밖의 문장부호 뒤에 위치해야 한다.

숫자 대신 기호 문자나 임의의 텍스트를 각주표지로 쓰고 싶다면 `\footnote` 명령에 관련된 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.

이런 방식으로 각주 번호를 로마 숫자나 멋진 기호문자로 바꿀 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[II] is the first footnote, and this[II] is the second. The end.[†]

[II] The first.

[II] The second.

[†] At last!

`\fnsymbol{footnote}`에 주목하자. 이것은 아홉 개의 기호문자를 1에서 9까지의 각주 번호 카운터에 할당한다. * † ‡ § ¶ ‖ ** †† ‡‡

동일한 각주에 대해서 여러 번 참조하려면 각주 숫자를 하나하나 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.
```

This[1] and this[2] are footnotes: please read note[1] again.

[1] the first.

[2] the second.

주의: minipage는 그 자체의 각주 번호 카운터를 별도로 가진다. 각각 `mpfootnote`와 `thempfootnote`이다.

각주에 관한 문제는 제법 복잡해서, 복잡한 각주, 둘 이상의 번호 체계를 갖는 복합각주, 각주 안의 verbatim, 각주 안의 수식, parbox 안의 각주, 페이지별 각주번호 등의 문제를 해결하는 패키지들이 많이 있다. 이럴 경우 예를 들면 footnote나, footmisc 등의 사용설명서를 참고해야 한다. 그러나 LaTeX에서 각주 문제는 거의 대부분이 해결되어 있으므로 문제는 어떤 패키지를 참고해야 하느냐일 뿐이다. 가장 좋은 방법은 질문하는 것이다.

문서 끝의 각주(미주)

endnotes 패키지는 모든 각주를 문서 끝으로 몰아준다. preamble에 다음 한 줄을 추가해야 한다.

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

그리고 다음 몇 줄을 문서의 마지막에 둔다.

```
\newpage
\begingroup
\parindent 0pt
\parskip 2ex
\def\enotesize{\normalsize}
\theendnotes
\endgroup
```

이 이외의 다른 명령도 있다. `endnotes.sty` 소스 파일을 읽어보라.

4.6 Insert/Indices 삽입/목차

차례, 표 차례, 그림 차례를 생성하고 넣는 것은 LaTeX에서 아주 간단한 일이다. 다음 몇 줄을 첫번째 `\section`이나 `\chapter`보다 이전에 써주기만 하면 된다.

```
\tableofcontents
\listoffigures
\listoftables
```

[역자 추가] 목차의 점선이나 페이지 표시 방법 등을 사용자가 수정하려면 tocloft 패키지를 이용할 수 있다. 그리고 float 패키지를 이용하여 새로운 플로트를 정의할 때 그 목록을 만들 수도 있다. 이 두 가지 패키지는 모두 memoir에 이미 들어 있다.

4.7 Insert/Vertical and Horizontal Space 삽입/수직 수평 간격

이 항목이 존재하는 워드 프로세서는 내가 아는 한 없다. LaTeX은 이 일을 아주 우아한 방식으로 처리한다. 그러나 워드 프로세서는 여기에 대응하는 기능이 없다.

공백 채우기는 텍스트를 수직으로 수평으로 가운데 두기 위해 사용한다. 워드 프로세서에서는 이것이 매우 어려운 일이다. 여러 번 시행착오를 거쳐야만 겨우 비슷하게 된다. `\null`이나 `~`를 고정점으로 사용하고 그 뒤에 이어 `\vfill`이나 `\hfill`을 다음 보기와 같이 써보라.

one two

three

four five

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

보통 LaTeX은 사용자가 마음대로 빈 공간을 넣는 것을 허락하지 않는다. 소스에서 스페이스를 두 번 친다고 해서 출력물에서 두 칸의 스페이스가 나타나는 것은 아니다. 그러나 문서가 엉망이 되어도 상관없다면 잘라지지 않는 공백 기호 ~(tilde)를 두 번 써보라. 실제로 출력에도 두 개의 공백이 찍힐 것이다.

이런 식으로 하지 않고 텍스트를 페이지의 절대 위치에 고정시키고 싶은 경우에는 4.11절에서 언급하고 있는 textpos를 이용하는 방법이 있다.

또, `\hspace`를 다음과 같이 사용할 수 있다.

This is a 2-cm-wide hole.

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

[역자 추가] `\hspace`는 '앞 글자'가 있어야 동작한다. 왼쪽 끝에서 `\hspace`는 아무 의미가 없을 것이다. 이럴 때도 강제로 간격을 주려 한다면 `\null\hspace`와 같이 하거나 또는 '별표 붙은 명령' `\hspace*`를 쓸 수 있다.

4.8 Insert/Tabs 삽입/탭

tabbing 환경은  키의 동작과 거의 비슷한 기능을 제공한다. 다음 보기를 보라.

Zero	One	Two	Three
Zero	One		Three
	Zero	Two	Three
Zero	One	Two	
new tab 1···	new tab 2		
new	tab		
Zero	One	Two	Three

```
\begin{tabbing}
% let's set the tab positions
~ \hspace{1cm} \= ~ \hspace{1.5cm} \=
~ \hspace{2.5cm} \= \kill % discard text
Zero \> One \> Two \> Three \\
Zero \> One \> \> Three \+ \\ % go right
Zero \> Two \> Three \- \\ % go left
Zero \> One \> Two \\
\pushtabs % save tab positions
new tab 1{\ldots} \= new tab 2 \\
new \> tab \\
\poptabs  % restore tab positions
Zero \> One \> Two \> Three
\end{tabbing}
```

tabbing 환경이 일부 불편한 점이 있는 데 비해 최근 tabto 패키지는 더 편리하게 tab을 이용한 정렬을 할 수 있게 해준다. 특히 특별히 특정 환경으로 둘러싸지 않아도 tab을 쓸 수 있다는 것이 장점이다. 해당 패키지의 문서를 참고하라.

또 tabular와 table 환경을 참고하라.

4.9 Insert/Cross Reference 삽입/교차참조

\label, \ref, \pageref 명령만 있으면 텍스트에 레이블을 달고 그것에 대해 교차참조 할 수 있다. 레이블의 표준 포맷은 prefix:suffix 꼴인데, prefix는 cha(장), eq(수식), fig(그림), sec(절), tab(표)와 같은 것이 될 수 있다. 이와 같이 prefix를 붙이는 것은 소스를 좀더 알아보기 쉽게 만들려는 것으로 꼭 이런 형식이어야만 하는 것은 아니다.

절 번호, 표 번호, 그림 번호, 페이지 번호 등을 참조하려면 \label과 \ref 명령을 다음 보기와 같이 사용한다.

$\mathcal{AMS}$-math는 수식 번호를 괄호로 감싸주는 \eqref라는 명령을 별도로 제공한다.

Example. This paragraph appears in Section 4.9 on page 16.

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

물론 사용자가 prefix를 마음대로 정할 수도 있다. 다음과 같은 enumerate 리스트를 생각해보자.

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

```
\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}
```

4.10 Insert/Margin Notes 삽입/방주

정말 쉽다. \marginpar{text}.

4.11 Insert/Frame 삽입/프레임

포스터나 광고판을 조판한다고 생각해보자. 특정 텍스트나 그림을 페이지상의 고정된 위치에 갖다두어야 할 때가 있다. 이럴 경우 textpos패키지를 이용한다. 샘플이 그림 A.5에 있다. (부록 A를 보라.)

유명한 레이아웃 프로세서에서 프레임이란 페이지 상에 놓이는 특정의 구역을 말한다. 이것은 워드 프로세서에는 없는 개념이다. LaTeX으로 이와 같은 레이아웃 프로세서의 프레임을 구현한 예로는 flowfram 패키지가 있다. 이것은 이미 텍스트의 흐름을 조판하는 것이 아니므로 대부분의 문서에서는 볼 수 없는 것이다.
한편, 프레임을 단순히 박스쳐진 텍스트로 부르는 경우도 있는데 이것은 framed나 boxedminipage, 또는 boites 패키지에 의하여 구현이 가능하다. 특히 framed, boites는 페이지 사이가 나누어지는 프레임쳐진 문단을 식자할 수 있게 해주기도 한다. 이에 대해서는 5.4절을 참조.

더 쉬운 접근은 *minipage*를 사용하는 것이다. miniature page라는 뜻이다. minipage 환경 안에는 텍스트, 그림, 어떤 것이라도 올 수 있다. 나아가 boxedminipage 패키지가 제공하는 boxedminipage 환경도 있다. 이름 그대로 minipage에 박스 테두리를 그려준다. 이 문단은 다음과 같이 선언된 것이다.

```
\begin{boxedminipage}[c]{0.6\linewidth}
... text ...
\end{boxedminipage}
```

4.12 Insert/Figure 삽입/그림

(LaTeX에서 그림을 포함하는 문제에 대한 안내서로 'Using Imported Graphics in LaTeX 2ε', a.k.a. epslatex.ps가 있다.)

좀 오래된 안내서이고 요즘 사정과 달리 eps 그림에 대해 집중적으로 설명하고 있지만 훌륭한 문서.

'figure'라 함은 비단 그림 파일만을 의미하는 것이 아니라 텍스트, 표 등 figure 환경 안에 놓을 수 있는 것은 뭐든지 상관없다. 다음 보기를 보자.

```
\begin{figure}[htbp]
% [htbp] specifies the
% preferred placement: here, top,
% bottom, or separate page.
  \begin{center}
  \texttt{=8-)}
  \end{center}
  \caption{A smiley representing
  the author of this guide.}
  \label{fig:mysmiley}
\end{figure}
```

=8-)

그림 1: A smiley representing the author of this guide.

그림들이 figure 관련 코드를 작성한 바로 그 위치에 정확하게 나타난다는 보장이 없음에 주의하자. 사실 워드 프로세서와 LaTeX의 가장 중요한 차이 중 하나가 그림들이 고정된 위치를 갖지 않는다는 점이다. 그림은 LaTeX이 스스로 결정하는 최적의 위치로 '떠다닌다'. 그러므로 문장을 쓸 때는 '아래 그림'이나 '위의 그림'과 같이 써서는 안 되고 '그림~\ref{fig:label}'과 같이 작성해야 한다. 그림이 어느 위치에 올지 모르기 때문이다.

사실 LaTeX 처음사용자가 가장 짜증스러워하는 것이 표나 그림이 "제 위치"에 오지 않는다는 것이다. 그러나 다시 생각해보면, 과연 사용자가 원하는 그 위치가 반드시 최적의 위치일까? 그림이 다음 페이지로 넘어가고 앞 페이지의 아래쪽이 휑하게 비는 것이 꼭 좋은 위치인 것일까? 이것은 짜증을 부릴 일이 아니라 왜 LaTeX이 표나 그림을 그런 식으로 처리하는지 이해하는 것이 더 낫지 않을까 싶다. 만약 그래도 표/그림의 위치를 LaTeX에게 맡기기 싫다면, 그냥 float 환경 안에 넣지 않으면 된다.

이런 속성 때문에 그림이나 표를 **떠다니는 개체**라고 부른다. 특정의 표나 그림이 정확하게 어떤 위치에 있어야 할 이유가 꼭 있다면, here 패키지를 사용하라. 이 패키지는 위지 지정인자로 H를 제공한다.

Encapsulated PostScript (.eps) 포맷의 그림이 하나 있다고 하자. 이 그림을 LaTeX 소스 파일에 삽입하려면 graphicx 패키지와 그림 2에 보인 것과 같은 명령을 이용한다.

```
\begin{figure}
\begin{center}
\fbox{\includegraphics
[width=0.5\textwidth, angle=-90]
{gnuplot}}
\caption{A Gnuplot graph.}
\label{fig:gnuplot}
\end{center}
\end{figure}
```

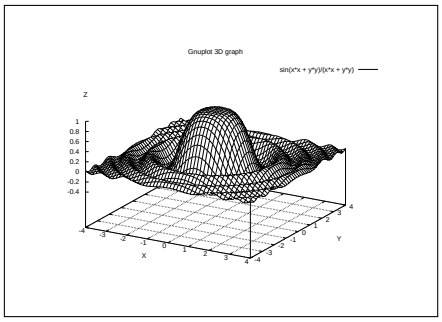


그림 2: A Gnuplot graph.

latex과 dvips로 문서를 조판할 때는 EPS 파일만이 동작한다. 반면 pdflatex은 JPG, PNG, (당연히) PDF 파일을 받아들인다.

TeX Live 2010을 기준으로 pdflatex도 EPS 파일을 손쉽게 처리한다. 따라서 pdflatex에서 .eps 파일을 다룰 때의 주의사항은 이제 해결된 문제가 되었다. 이것은 XeTeX도 마찬가지다.

일반 그래픽 포맷(.jpg, .gif, .png 등)을 .eps로 변환하는 패키지들이 몇 가지 있다. 예를 들면 ImageMagick (http://www.imagemagick.org), GIMP (http://www.gimp.org) 등. 그러나 이런 응용 프로그램들은 엄청난 크기의 PostScript 파일을 만들어낸다.

제일 좋은 것은 비트맵을 내장하여 컴팩트한 PostScript 파일을 만들어내는 응용 프로그램을 이용하는 것이다. jpeg2ps (http://www.pdflib.com/jpeg2ps/index.html), bmeps (CTAN://support/bmeps)와 같은 유틸리티가 좋다. 앞의 것은 .jpg 파일을 내장하는 데 제일 낫다고 하고, 뒤의 것은 다양한 그래픽 포맷을 지원한다.

현재, 그림 포맷과 관련해서는 다음과 같이 알아두면 된다. latex-dvips를 쓰는 경우에만 오직 .eps 그림이 필요하다. 그 외의 경우, PDFLaTeX, XeLaTeX, latex-dvipdfmx 등, .png, .jpg, .pdf, .eps 어떤 포맷이라도 상관없다. 다만 경우에 따라 \usepackage{epstopdf}를 지정해주어야 하는 때는 있다 (PDFLaTeX).

☞ PDFLaTeX을 사용할 때는 그림을 .pdf로 변환해야 한다. 이를 위해 epstopdf 프로그램을 명령행에서 쓰면 되고 이에 맞추어 소스를 수정해야 한다!

같은 소스에서 .pdf와 .ps를 모두 만들려 한다면 다음과 같은 코드를 포함하는 방법이 있다.

```
\usepackage{ifpdf}
...
% include the right options
\ifpdf
  \usepackage[pdftex]{graphicx}
  \pdfcompresslevel=9
\else
  \usepackage{graphicx}
\fi
...
% include the right graphic file
\ifpdf
  \includegraphics{file.pdf}
\else
  \includegraphics{file.eps}
\fi
```

XeTeX과 같은 엔진을 사용할 때는 소스가 좀 달라져야 한다. 이 샘플은 pdflatex과 latex-dvips 두 가지 경우만을 상정하고 있다.

한글 문서를 XeTeX 용으로 작성하는 경우의 예를 들어보자.

```
\usepackage{iftex}
\ifXeTeX
  \usepackage{graphicx}
\else\ifPDFTeX
  \usepackage{graphicx}
```

```
  \usepackage{epstopdf}
\else
  \usepackage[dvips]{graphicx}
\fi\fi
...

\ifXeTeX
  \includegraphics{file}
\else\ifPDFTeX
  \includegraphics{file}
\else
  \includegraphics{file.eps}
\fi\fi
```

☞ 18개 이상의 플로트가 처리되지 않은 상태로 대기중이면 'Too many unprocessed floats'라는 에러를 만나게 된다. 이 문제를 해결하는 가장 빠른 방법은 \clearpage를 서너 개의 그림 사이에 넣어주는 것이다. 또는 morefloats 패키지를 이용할 수도 있다.

문단을 파고드는 그림

잡지 등의 레이아웃에서 볼 수 있는 그림이 텍스트 문단을 파고 들어가는 것은 wrapfig 패키지를 이용한다.

If you meet this guy, give him some money.

=8-) The reason may not be apparent 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.

\begin{wrapfigure}[4]{l}[5pt]{2cm}
{\Huge
 \texttt{=8-)}
}
\end{wrapfigure}

The reason may not be apparent 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?
```

인자는 좁아지는 문단의 행수, 그림의 위치, 그림 걸이 길이(overhang), 그림의 폭(width)이다.

4.13 Insert/Shapes 삽입/그림마당

LaTeX 자체가 picture 환경을 제공하고 있다. \circle, \oval 등의 명령을 이용해서 그림을 그릴 수 있다. 내 생각에 그림그리기 환경 없이 뭔가를 그린다는 것은 너무 어렵고 picture 환경은 자체의 한계를 몇 가지 가지고 있기도 하다. 차라리 Xfig (http://www.xfig.org)와 같은 드로잉 프로그램을 이용하는 것이 훨씬 낫다. 이 프로그램은 Unix에서만 사용할 수 있고 몇 가지 지적할 만한 훌륭한 기능을 가지고 있다.

Xfig는 겉모양이 별로다. 그러나 매우 강력하다. 멋진 장점 중의 하나는 여러 포맷으로 드로잉을 내보내기할 수 있다는 점이고 그 가운데는 LaTeX이 이해하는 포맷이 포함된다.

xfig와 매우 비슷한 환경을 제공하는 platform-free java 드로잉 툴인 jfig (http://tams-www.informatik.uni-hamburg.de/applets/jfig/)가 있다. Windows에서 간단한 드로잉을 원한다면 TpX (http://tpx.sf.net)라는 프로그램도 있다.

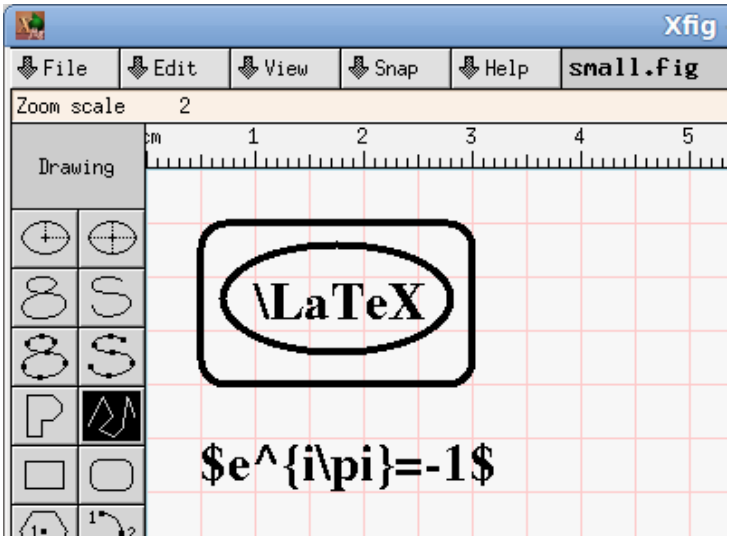


그림 3: Xfig로 만든 드로잉

또다른 장점으로 'special flag' 필드가 세트되면 텍스트 오브젝트가 LaTeX으로 렌더링된다는 것이다. 이것은 드로잉 안에 일반적인 TeX 수식을 넣을 수 있다는 뜻이 된다.

따라서, `small.jpg`라는 드로잉(그림 3)을 만들었다고 해보자. `pdflatex`을 사용할 것이라면 File/Export…를 열어서 'Combined PDF/LaTeX (both parts)' 항목을 선택하자. 그러면 Xfig는 두 개의 파일을 만들어주는데, 하나는 `small.pdf`이고 다른 하나는 `small.pdf_t`이다. 문서에 이 드로잉을 포함하려면 다음과 같이 한다.

```
\usepackage[pdftex]{graphicx}
...
Here's an Xfig drawing:

\input{small.pdf_t}
```

Here's an Xfig drawing:

LaTeX

$e^{i\pi} = -1$

플레인 `latex`을 사용할 것이라면 File/Export…를 열어서 Language 메뉴에서 'LaTeX picture + eepic macros'를 선택한다. 그러면 `small.eepic`라는 파일을 얻을 수 있다. 문서에 그림을 포함하려 할 때 epic과 eepic 패키지를 사용해야 한다.

```
\usepackage{epic}
\usepackage{eepic}
...
This is a picture
drawn with Xfig:\\
\input{small.eepic}
```

주의할 점이 있다. epic와 eepic는 hyperref과 충돌이 있는 것 같다. hyperref이 `\path` 명령을 재정의하기 때문이다. `pdflatex`을 써야 할 또하나의 이유이다.

진짜 멋진 그림을 그리고 싶다면 pgf `http://sourceforge.net/projects/pgf/` 패키지나 pstricks `http://tug.org/PSTricks/main.cgi`를 고려해보라. 이 패키지들은 LaTeX으로 멋진 PostScript 드로잉을 그릴 수 있게 해준다. 또다른 괜찮은 프로그램으로 ePix

pgf와 pgf의 인터페이스 명령 집합인 ti*k*z는 beamer의 기본 드로잉 도구이다. 또한 ti*k*z를 이용한 여러 유틸리티들이 개발되어 있다. pstricks는 매우 강력하지만 현재 그 능력을 충분히 발휘하려면 latex-dvips 루트를 거치지 않으면 안 된다는 점이 한계로 작용하고 있다. 문서를 풍부하게 할 드로잉 도구가 필요하다면 먼저 pgf/ti*k*z를 검토해보는 것을 추천한다.

http://mathcs.holycross.edu/~ahwang/current/ePiX.html라는 것도 있는데 이것은 LaTeX 문서 안에 포함할 학문적인 플로트나 그림을 생성해주는 데 특화하고 있다. 이 밖에도 많은 도구가 있다. 웹을 "LaTeX vector graphics"로 검색해보라.

4.14 Insert/Line 삽입/선

임의의 길이와 굵기로 된 선분을 그리는 명령은 \rule이다.

This is a page-wide rule:

but this one is shorter and thicker:

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

\dotfill은 점으로 만들어진 재미있는 '선'을 그어준다. 이것은 서로 연관된 것들을 이어주는 데 가끔 쓰인다.

Total price .....€ 10

```
Total price \dotfill \euro~10
```

4.15 Insert/Hyperlink 삽입/하이퍼링크

url 패키지는 URL 주소를 쓰고 하이프네이션을 적용해준다. hyperref 패키지와 함께 사용하면 dvipdf, pdflatex을 통해서 하이퍼링크가 동작하는 .pdf 문서를 만들 수 있다. 예를 들면 이 문서는 다음과 같이 선언하여 만들어진 것이다.

hyperref 패키지와 url을 별도로 로드할 필요는 없다. hyperref 패키지가 url의 역할을 모두 한다.

```
\usepackage[colorlinks,urlcolor=blue,filecolor=magenta]{hyperref}
%\usepackage{url}
```

예를 하나 들어보자.

The CTAN main site is http://www.ctan.org, a.k.a CTAN://.
Listen to this MIDI file.
Click here to go back to the top.

```
The \hypertarget{ctan}{CTAN} main site
is \url{http://www.ctan.org}, a.k.a
\href{http://www.ctan.org}{CTAN://}.

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

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

\hypertarget과 \hyperlink 명령은 HTML에서와 같은 내부 링크를 만들어준다. \href는 URL이나 외부 파일에 대한 링크를 만든다. run: 파라미터에 주의하라. 이것은 멀티미디어 플레이어, 오피스 프로그램 등과 같은 외부 프로그램을 실행시켜준다. 내가 아는 한 이 기능은 오직 Adobe Reader, Okular, Evince에서만 동작한다.

Linux나 다른 Unix 계열 운영체제에서는 외부 파일이 참조되었을 때 어떤 PDF 리더를 실행해야 할지를 지정해주어야 한다. 자신의 .mailcap이나 /etc/mailcap에 다음 내용을 써넣으면 된다.

```
audio/midi;/usr/bin/timidity %s
audio/*; xmms %s
video/*; xine -pfhq %s
```

예를 들면 외부 파일의 타겟을 참조하거나, pdf bookmark를 만들거나 Adobe Acrobat의 메뉴를 조작하는 등의 작업도 가능하다.

hyperref의 패키지 문서를 읽어보면 더 많은 예와 기능을 알 수 있다.

4.16 Insert/Comment 삽입/주석문

각 행 앞에 % 기호를 붙이면 그 행은 주석문이 되어 문서의 출력에 반영되지 않는다. 여기에 출력물에는 보이지 않는 저자 자신의 메모나 노트를 기록할 수 있다. 또는 comment 패키지를 통하여 `comment` 환경을 쓰면 문서의 일부를 무시하게 만들 수 있다.

5 Format 모양 메뉴

일반적으로 문서의 주요 포맷 설정은 `\documentclass`의 파라미터로 지정한다. 기본 글꼴 크기(10, 11, 12pt), 용지(`a4paper`, `a5paper`, `b5paper`, `letterpaper`, `legalpaper`, `executivepaper`), 방향(`portrait`, `landscape`) 등.

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

위의 세 가지 이외의 폰트 크기를 지정하는 것도 가능한데 이에 대해서는 5.2절에서 설명한다.

5.1 Format/Line Spacing 모양/줄간격

memoir에는 setspace의 기능이 그대로 들어 있으나 명령과 환경의 첫 글자가 대문자로 시작하는 것이 다르다. 예를 들면 `\Spacing`.

setspace 패키지는 `singlespace`, `onehalfspace`, `doublespace` 환경을 제공한다. 그리고 `\spacing`*{amount}* 명령(환경)은 주어진 크기만큼 행간격을 설정해준다.

These two lines

are crazily spaced!

Much better, these lines
have a pretty space.

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

한글 문서는 기본값이 1.333

전체 문서의 행간격을 적용하려면 `\linespread{factor}` 명령을 preamble에서 사용한다. 기본값은 `factor` = 1로 되어 있다. 이 값이 커지면 행간격도 커진다. 1.6이면 대략 double line spacing에 해당한다.

5.2 Format/Character 모양/글자

표준적인 글자 속성은 표 3에 열거하였다. 글자 크기는 표 4를 보라.

이탤릭체와 강조체 사이의 차이에 주의하라. `\emph` 명령은 이탤릭체 안에서는 upright 체로 변한다. *For example, this portion of text is typeset in italics, and* these words *are emphasised in upright*. `\emph` 명령은 타이포그래피 지정 명령이 아니라 '강조'를 의미하는 논리적인 명령임을 알 수 있다.

또한 아래첨자는 수학 모드 안에서만 동작한다는 것도 알아두자. 일부 `\textsubscript`가 정의되어 있는 클래스를 쓴다면 이를 이용하여 `this is \textsubscript{subscript}`

Text attribute	Environment form	WBoth
\textnormal	textnormal	main document font
\textrm	rmfamily	roman
\textit	itshape	*italics*
\emph	n/a	*emphasis*
\textmd	mdseries	medium weight (default)
\textbf	bfseries	**boldface**
\textup	upshape	upright (default)
\textsl	slshape	*slanted*
\textsf	sffamily	sans serif
\textsc	scshape	SMALL CAPS
\texttt	ttfamily	typewriter
\underline	underline	underline
\textsuperscript	n/a	this is $^{\text{superscript}}$
\mathrm	n/a	$\mathrm{x^n + y^n \neq z^n \forall n \neq 2}$
\mathbf	n/a	$\mathbf{x^n + y^n \neq z^n \forall n \neq 2}$
\mathsf	n/a	$\mathsf{x^n + y^n \neq z^n \forall n \neq 2}$
\mathtt	n/a	$\mathtt{x^n + y^n \neq z^n \forall n \neq 2}$
\mathit	n/a	$\mathit{x^n + y^n \neq z^n \forall n \neq 2}$
\mathnormal	n/a	$x^n + y^n \neq z^n \forall n \neq 2$
\mathcal	n/a	$\mathcal{\S\ + \dagger\ \neq \ddagger\forall\ \neq \in}$

표 3: 글꼴 속성

Font size	WBoth
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 4: 폰트 사이즈

로 입력하여 this is$_{\text{subscript}}$와 같이 쓰는 것이 가능하지만 그렇지 않을 경우 노멀 텍스트에서 첨자를 사용하는 트릭은 다음과 같다.

this is subscript

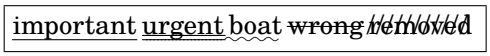
```
this is
$_{\mbox{\footnotesize{subscript}}}$
```

\textsuperscript 명령을 통해서 윗첨자는 텍스트 모드에서도 쉽게 식자할 수 있다. \textsubscript 명령은 별도로 정의된 패키지가 사용되어야 하는데, memoir 클래스에는 이 명령이 미리 정의되어 있다.

밑줄 긋기

일반적으로 밑줄 긋기는 사용하지 않는다. 이것은 옛날 텔레타이프 시대의 유물로서 보기에 좋지도 않다. 정말로 밑줄을 긋지 않으면 안 되는 경우, 보통은 \underline 명령으로 처리할 수 있지만 밑줄 그어진 단어 단위가 행자름이 되지 않는 불편이 있다. 이런 문제를

해결하고 몇 가지 더 재미있는 밑줄 스타일을 제공하는 ulem 패키지를 사용할 수 있다. 다음 보기를 보자.




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

주의 : ulem은 \emph 명령을 재정의하여 밑줄 긋기로 바꾼다. 이것을 피하려면 패키지를 다음과 같이 불러야 한다.

이 문서를 HLaTeX으로 처음 번역할 때 이 ulem 명령 관련된 부분이 잘 되지 않아서 애를 먹었다. 그래서 myulem을 별도로 만들어야 했었다. 지금은 너무나 잘 되기 때문에 격세지감을 느끼지 않을 수 없다.

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

Format/Character Size 모양/글자 크기

표준 폰트 사이즈로 충분치 않다면 extsizes가 도움이 된다. 표준 문서 클래스의 글자 크기 옵션을 '확장'하여 8–12, 14, 17, 20pt 옵션을 추가해준다.

예를 들어 어떤 글을 본문 17포인트로 조판하기를 원한다고 하자. preamble에 다음과 같이 쓰면 된다.

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

큰 글자를 얻는 또다른 방법은 type1cm 패키지를 이용하는 것이다. 다음과 같은 명령이 가능하다.

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

(이 샘플은 너무 커서 이 페이지에 맞추기가 어렵기 때문에 결과를 보이지 못한다.)

인자 두 개는 각각 폰트의 사이즈와 베이스라인스킵의 크기이다. 또다른 방법은 다음과 같은 것이다.


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

Format/Character Font 모양/폰트

이 부분은 XeTeX이 등장하기 전 legacy TeX의 폰트와 LaTeX 2ε의 NFSS에 대해서 기술하고 있다. XeTeX의 등장으로 사정이 무척 많이 달라졌다. 이에 대해서 역자가 말미에 짧은 한 문단을 추가한다.

LaTeX은 자체 폰트를 사용하는데 필요하면 자동으로 METAFONT 시스템에 의하여 생성한다. 이렇게 하면 이식성이 보장되며 매우 좋은 출력 품질을 얻을 수 있다. 그러나 우리는 Times, Helvetica, Sans Serif…와 같은 다른 폰트에 익숙하다.

다행히 LaTeX은 PostScript 폰트를 사용할 수 있다. 다음 패키지들은 익숙한 PostScript 폰트를 사용할 수 있게 하는 것들이다. avant, avangar, bookman, chancery, charter, courier, helvet, helvetic, ncntrsbk, newcent, palatcm, palatino, pifont, times, utopia, zapfchan. 문서에 \usepackage{times}라고 써넣고 결과를 살펴보라. 유일한 주의사항은 LaTeX이 수식을 다룰 때 Computer Modern보다 나은 폰트가 없다는 점이다. PostScript 폰트를 수식에 썼을 때는 품질이 좀 떨어지는 것으로 보일 수도 있다.

Family	Name
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 5: 일반적인 폰트 패밀리

위에 열거한 패키지들은 문서 전체의 폰트를 설정한다. PostScript 폰트를 텍스트 영역에만 적용하고자 한다면 폰트 패밀리를 아래 예와 같이 지정해주도록 하라. 일반적인 폰트 패밀리를 표 5에 열거하였다.

☞ 주의. 몇몇 폰트는 시스템에 따라 이용불가능할 수 있다.

This is Computer Modern Roman, this is Helvetica!

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

다른 가능성은 표준 LaTeX 폰트를 PostScript 폰트로 교체하는 것이다. 예를 들면 Computer Modern Sans Serif가 나타날 위치의 폰트를 전부 Avantgarde로 바꾸는 것이다. 재정의 가능한 명령은 다음과 같고,

- \rmdefault (roman)
- \sfdefault (sans serif)
- \ttdefault (typewriter)
- \bfdefault (boldface)
- \mddefault (medium)
- \itdefault (italics)
- \sldefault (slanted)
- \scdefault (small caps)
- \updefault (upright)

sffamily 기본 글꼴을 Computer Modern Sans Serif로부터 Avantgarde로 바꾸는 것은 다음과 같이 한다.

```
 % Avantgarde replaces sans serif
\renewcommand{\sfdefault}{pag}
```

[역자] 위의 예를 XeTeX에 적용한 것. 폰트의 모양이 제대로 나오는 예이다. 이 문서는 XeTeX으로 만들고 있으므로 위의 예에서는 폰트의 모양이 실제 모양으로 나오지 않았다. XeTeX에서는 `\fontfamily`와 같은 명령 대신 `\fontspec`을 쓴다. 여기서는 Computer Modern Roman의 글자 모양을 보이기 위해서 lmroman을 썼는데 lmroman이 cmr를 바탕으로 만들어진 폰트이므로 글자 모양을 확인하는 데는 지장이 없을 것이다.

This is Computer Modern Roman, This is Helvetica!

```
\fontspec[ExternalLocation]{lmroman10-regular}
This is Computer Modern Roman,
\fontspec{Helvetica}
This is Helvetica!
```

XeTeX은 legacy TeX의 폰트 사용 방법에 일대 혁신을 가져왔다. Jonathan Kew가 제작한 XeTeX은 기존의 TeX 시스템에서 폰트를 처리하는 부분을 유니코드 폰트 사용 방식으로 교체한 것이다. 그 결과 운영체제에 설치된 오픈타입, 트루타입 폰트를 TeX에서 즉시 사용할 수 있게 되었다.

Format/Character Colour 모양/글자 색

xcolor를 이용하면 훨씬 많은 "이름으로 정의된 색상"을 사용할 수 있다.

color 패키지를 이용하면 색상 이름과 적절한 명령을 사용할 수 있다. 미리 정의된 색상은 black, white, red, green, blue, cyan, magenta, yellow이다. 자신의 색상을 정의하는 것도 가능하다.

This is red.
This text is blue!
So is this. Let's change.
This is my shade of green!
A cyan box
A green box in a blue frame

```
\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}
```

이따금 페이지 전체에 배경을 깔고 싶을 때가 있다. 전에는 이 절차가 무척 복잡했지만 wallpaper 패키지로 매우 간단해졌다. 그림을 먼저 준비하고 이 패키지를 이용하면 손쉽게 배경에 그림을 놓을 수 있다.

`\pagecolor`라는 명령도 있다. 이 명령을 쓰면 어떤 일이 일어날까?

5.3 Format/Paragraph 모양/문단

LaTeX에 있어서 문단이란 무엇을 의미하는 것인지 상기하자. 문단이란 `\\`나 빈 줄로 끝나는 텍스트의 일부이다.

자꾸 지적하는 것이지만, `\\`는 문단끝을 나타내는 기호가 아니다. 저자는 문단에 대해서 오해를 하고 있는 것 같은데, `\\`는 행의 줄바꿈을 강제하는 것일 뿐이고 문단을 끝내고 새 문단을 시작하는 것이 아닌 것이다. 문단 끝은 빈 줄 또는 `\par`가 오면 끝난다. 당연히 `\\` 다음에 오는 빈 줄은 문단 첫 줄이 가지는 특성(들여쓰기 등)을 가지지 않는다.

환경 *(environments)*이라 하는 것은 텍스트의 일부분에 대해 정렬(alignment)이나 글꼴 선택과 같은 특정 속성을 부여하는 LaTeX의 방식이다. 이것은 마우스로 텍스트를 선택한 다음에 원하는 속성을 메뉴나 버튼 클릭으로 부여하는 것과 비슷하다. 중괄호로 텍스트 일부를 둘러싸는 것도 영역 선택과 비슷한 효과를 갖는다.

환경의 일반적 형태는 다음과 같다.

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

예를 들어 어떤 문단을 가운데 정렬하고 싶으면 `center` 환경을 사용한다.

this text is centered

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

표준적인 환경들을 표 6에 열거하였다. 다음 절에서 언제 어떤 환경을 사용하는지 예를 들어보겠다.

Environment	**Purpose**
`array`	Math arrays
`center`	Centered lines
`description`	Labelled lists
`enumerate`	Numbered lists
`eqnarray`	Sequence of aligned equations
`equation`	Displayed equation
`figure`	Floating figures
`flushleft`	Flushed left lines
`flushright`	Flushed right lines
`itemize`	Bulleted lists
`letter`	Letters
`list`	Generic list environment
`minipage`	Miniature page
`picture`	Picture with text, arrows, lines and circles
`quotation`	Indented environment with paragraph indentation
`quote`	Indented environment with no paragraph indentation
`tabbing`	Align text arbitrarily
`table`	Floating tables
`tabular`	Align text in columns
`thebibliography`	Bibliography or reference list
`theorem`	Theorems, lemmas, etc
`titlepage`	For hand crafted title pages
`verbatim`	Simulating typed input
`verse`	For poetry and other things

표 6: 표준적인 LaTeX 환경

Format/Paragraph Horizontal Alignment 모양/문단 정렬

정렬 방식의 더 정밀한 조절이 필요하거나 (이따금 그러하듯이) `\justifying`이 필요하다면 ragged2e 패키지를 사용한다.

텍스트는 좌우정렬되는 것이 기본이다. 왼쪽정렬, 오른쪽정렬, 가운데정렬 하기를 원할 때는 `flushleft`, `flushright`, `center` 환경을 쓴다. `\raggedright`, `\raggedleft`, `\centering`은 순서대로 각각의 환경에 대응하는 명령들이다. 그러나 이 명령은 새 문단을 시작하지 않는다.

Format/Paragraph Vertical Alignment 모양/문단 사이 띄우기

문단 사이가 벌어지는 방식은 워드 프로세서 사용자를 종종 당황하게 만든다. **여러 개의 빈 줄과 여러 개의 스페이스는 한 개의 빈 줄이나 한 개의 공백과 똑같이 취급된다.** 따라서 소스에서 여러 개의 빈 줄을 넣는다고 해서 그만큼 문단 사이에 간격이 늘어나는 것이 아니다. 문단 사이 간격을 강제로 벌리려면 `\smallskip`, `\medskip`, `\bigskip` 명령을 사용해야 한다.

더 넓은 간격이 필요하다면 \vskip 명령을 다음 보기와 같이 사용한다.

These paragraphs will be separated by 1.3 cm:

there is a 1.3 cm gap above me.

```
These paragraphs will be
separated by 1.3 cm:\\
\vskip 1.3cm
there is a 1.3 cm gap above me.
```

이 글에서는 \vspace 명령에 대해서 기술하고 있지 않다. \vspace는 간격을 중괄호로 둘러싸서 인자로 지시하지만 \vskip은 중괄호 없이 그냥 길이를 지정한다. 그리고 \vspace에는 별표붙인 명령 \vspace*가 있어서 페이지가 바뀌더라도 수직 간격 명령이 동작하게 할 수 있다.

\vskip 명령은 문단 사이에서만 동작한다는 것을 알아두자. 따라서 이전 문단이 없는 페이지의 제일 윗쪽에 이 명령이 오더라도 아무런 의미가 없다. 새 페이지를 시작하고 추가로, 예컨대 1.5cm를 남기고 싶을 때는 어떻게 할 것인가? 이럴 때는 \null 명령을 이용해서 텍스트에 '표지'를 설정한다.

This text comes after 1.3 cm...

```
\null
\vskip 1.3 cm
This text comes after 1.3 cm...
```

마지막으로 \vfill 명령은 두 문단 사이에 가변적인 빈 공간을 넣어서 두번째 문단이 페이지의 하단에 맞추어지도록 한다. 예를 들면,

LaTeX에서 \vfil은 \vfill의 반(1/2)이다.

```
This appears at the top of
the page{\ldots}
\vfill
{\ldots}and this at the bottom.
```

This appears at the top of the page…

…and this at the bottom.

Format/Paragraph Margins 모양/문단 여백

quote 환경에 대한 언급을 역자가 추가

일반적으로 여백은 2.5절에서 본 바와 같이 문서 전체에 대해 설정되는 것이다. 일부 텍스트의 여백폭을 바꾸기 위해서 문서 중간에 이 파라미터 값을 바꾼다고 해도 동작하지 않는다. 한편 문단 폭이 달라지는 환경이 LaTeX에 이미 정의되어 있다. quote나 quotation 환경은 좌우 여백을 변경한다. 그런데 이런 환경을 쓰지 않고 임의로 문단 여백을 바꾸고 싶다면 다음 보기와 같이 새로운 환경을 만들어야 한다.

memoir와 changepage 패키지에는 문단 폭 변경을 위한 adjustwidth라는 환경이 있다. 흐르는 텍스트에는 대체로 잘 적용되지만 다른 리스트 환경과 함께 쓰면 원치 않는 동작을 하기도 한다.

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

그런 다음에 이 새로운 환경을 사용한다.

As you can see, this paragraph has normal margins.

But please note that this paragraph has custom margins.

```
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}
```

Format/Paragraph Indentation 모양/첫 줄 들여쓰기

문단 첫 줄의 들여쓰기 크기를 설정하려면 \parindent 명령에 적당한 값을 준다. 다음 예에서 첫 줄 들여쓰기 값을 1cm로 설정하였다.

LaTeX의 기본 값은 1.5em 정도를 들여쓰는 것이다. 우리 글 문서에서는 그냥 1em(한 글자 폭)을 선호하는 경우도 많다.

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

\indent 명령과 \noindent 명령은 해당 문단의 첫 줄을 들여쓰기 하게/못하게 강제한다.

문단 첫 줄을 들여쓰기 하지 않고 문단 사이에 추가적인 간격을 주어서 구분하는 경우도 있다. 이런 식으로 하려면 문단 사이의 추가 간격 길이인 \parskip 값을 정해주어야 한다.

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

5.4 Format/Paragraph Border and Shade 모양/문단 테두리와 음영

테두리쳐진 문단이나 단어를 얻으려면 \parbox 명령이나 framed 패키지를 써서 할 수 있다. \parbox를 쓸 때는 calc 패키지가 필요하다.

가장 쉬운 방법으로 framed를 쓰는 것은 다음과 같다.

this is a framed paragraph!

this is a shaded paragraph, do you like it?

```
\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}
```

마찬가지로 boxedminipage 패키지의 같은 이름의 환경을 쓴다. 더 자세히 알고 싶은 분을 위해 설명하자면 다음 명령

framed는 memoir에 이미 들어 있다. 별도로 \usepackage할 필요 없다. 페이지가 넘어가는 framed paragraph에는 boites 패키지도 고려해볼 만하다.

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

은 boxedminipage 환경과 동일하다.

다음 보기는 \parbox를 쓴 것이다.

again, a framed paragraph!

```
\noindent
\fbox{
  \parbox{.9\linewidth
    -2 \fboxsep -2 \fboxrule}
  {again, a framed paragraph!}
}
```

\linewidth는 minipage의 폭(width)를 글줄 길이와 같게 한다. 이 길이는 원하는 대로 지정해도 된다.

끝으로 텍스트 폭을 알아내어서 자동으로 테두리치도록 해보자.

this is a framed word

```
this is a
\framebox[\width]{framed}
word
```

파라미터를 조작하여 프레임의 폭을 조절할 수 있다.

this is another framed word

```
this is another
\framebox[2\width][r]{framed}
word
```

이 예에서 두 번째 선택 인자는 정렬(여기서는 right) 방식을 지정하는 것이다.

5.5 Format/Colour 모양/색상

소위 '형광펜 효과'로 soul 패키지의 \hl이 있다. 이것은 영문자의 경우 잘 동작한다. 한글에 대해서 이 비슷한 트릭을 만들어 본 예가 있다.

문단에 테두리를 치고 여기에 색을 입히려면 다음과 같이 한다.

I am a minipage, my colour is yellow!

```
\colorbox{yellow}{
  \begin{minipage}
  {0.8\linewidth}
  I am a minipage, my colour
  is yellow!
  \end{minipage}
}
```

칼러에 대해서 5.2절을 참조.

5.6 Format/Columns 모양/다단

multicols에 대해서는 저자의 착각인 듯. 패키지 이름은 multicol이고 환경의 이름이 multicols이다.

\twocolumn과 \onecolumn은 새 페이지를 시작하면서 정해진 다단을 설정한다. \documentclass 명령의 옵션으로 지정할 수 있다. 이것만으로 충분치 않다면 multicols 패키지가 같은 이름의 환경을 제공한다. 이 소절은 다음과 같은 명령으로 이단 편집하였다.

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

단 사이의 간격은 \columnsep 파라미터로 제어가능하고 \columnseprule 길이는 단 사이에 주어진 두께의 선을 그려준다.

\onecolumn이나 \twocolumn의 브래킷으로 둘러싼 선택 인자로 주어지는 텍스트는 해당 환경에서 제외된다. 예를 들어 \twocolumn에 일부 텍스트를 선택 인자로 주면 해당 부분을 1단으로 조판한 후에 2단을 시작한다.

5.7 Format/Styles 모양/스타일

이 소절은 역자가 추가한 것임

워드 프로세서 문서와는 달리 LaTeX은 처음부터 '구조화된' 문서를 만들게 되어 있다. 따라서 \chapter, \section, \subsection, \paragraph와 같은 장절 명령이 마련되어 있다.

보통 워드 프로세서에서 장절을 표현하기 위해서는 장절 타이틀에 해당하는 부분을 선택하여 글자 크기를 크게 하거나 두꺼운 글씨로 바꾸는 방법을 쓴다. 그리고 이렇게 만들어진 모양을 '스타일'로 등록하여 적용한다.

그렇다면 LaTeX에서 만약 \section 타이틀에 해당하는 모양을 바꾸고 싶다면 어떻게 하는가? 이 일을 해주는 패키지가 적어도 두 개 있다. 하나는 sectsty이라는 것이고 다

른 하나는 titlesec이라는 것이다. 예를 들어 sectsty을 사용하는 경우, section의 글자를 sffamily(산세리프체)의 Large 크기로 하고 싶다고 하자. 그러면 다음과 같이 한다.

```
\sectionfont{\sffamily\Large}
```

또 한 가지 주제로 section에 붙는 카운터의 모양을 바꾸는 문제가 있다. 이 카운터는 LaTeX이 자동으로 붙여주기 때문에 편하지만 모양을 바꿀 필요가 있기도 하다. 이것은 별도의 패키지를 사용하지 않고도 예를 들어

```
\renewcommand\thesection{\arabic{section}}
```

과 같은 방법으로 가능한데 이 재정의 이후에 수식 번호, 표 번호, subsection 번호 등이 자신의 의도대로 잘 바뀌는지를 살펴보아야 한다.

chngcntr라는 패키지가 이 일을 좀더 쉽게 해준다. 예를 들어

```
\counterwithout{section}{chapter}
```

와 같이 함으로써 section 카운터를 chapter 카운터로부터 분리할 수 있다. 이 때는 모양만 바꾸는 것이 아니라 chapter가 바뀌어도 section 번호는 이어서 계속된다. 이와 유사한 `\numberwithin|out` 명령을 $\mathcal{AMS}$-math 패키지가 제공해주고 있기도 하다.

위에 기술한 모든 기능을 memoir는 모두 제공하고 있으므로 별도의 패키지를 로드할 필요가 없다.

6 Table 표 메뉴

꽤 복잡한 주제……이다. *table*은 (4.12절에서 설명한) 떠다니는 개체로서 한 페이지 안에서만 존재한다. 즉 다음 페이지로 페이지가 나누어지지 않는 것이 원칙이다. 이 속에 들어가는 것은 대부분 tabular 환경인데 물론 다른 것도 얼마든지 올 수 있다. table의 폭은 그 내용물 길이에 따라 스스로 조절되는 것이 기본이다.

즉, tabular는 열과 행을 가지는 표 자체이고 table이란 이를 둘러싸서 떠다니게 해주는 조판을 위한 환경이다.

다음에 보인 것은 일반적인 테이블의 형식이다.

표(tabular)를 만드는 데 있어서 array 패키지를 빼놓을 수 없다.

```
\begin{table}[htbp] % placement: here, top, bottom, separate page
% \begin{small}     % sets the table font
\begin{center}      % optional, or use \centering command
% 4-column table; alignment is left, centered, right, fixed width
\begin{tabular}{|l|c|rp{4cm}|}
\hline              % horizontal line
\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}         % horizontal line spanning columns 1-2
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 \\
% force a space with "\ "
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}
\caption{A sample table.}
```

```
% labels are used for cross references;
% for example, "see Table~\ref{tab:sampletab}"
\label{tab:sampletab}
\end{center}
% \end{small}
\end{table}
```

표 7은 그 결과를 보여주고 있다.

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

표 7: A sample table.

이따금 table의 폭이 너무 넓어서 페이지에 맞지 않을 때가 있다. 이럴 경우 rotating 패키지가 `sidewaystable`이라는 새로운 환경을 제공해준다. rotating은 이밖에도 하나의 셀 내용을 주어진 각도만큼 회전시키는 것도 가능하다. 끝으로 tabularx 패키지는 전체 테이블이 일정한 폭을 갖도록 지정할 수 있다. `X` 컬럼지시자는 필요한 만큼 컬럼의 길이를 계산해서 늘려주도록 하는 데 사용된다.

비교적 최근에 나온 hvfloat나 floatrow 패키지들은 회전된 표를 만드는 다양한 방법을 제공한다. 또한 tabulary 패키지는 tabularx를 확장한 기능들을 제공한다.

예를 하나 들어본다.

```
\begin{sidewaystable}
  \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{sidewaystable}
```

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

이 문단은 역자가 추가하였음.

가끔 편집자들은 세로 괘선이 조판상 그다지 보기 좋지 않기 때문에 가로 괘선만으로 그리라고 요구하는 경우가 있다. 만약 세로 괘선이 전혀 없다면 booktabs 패키지를 이용하여 더 세련된 표를 만드는 것이 가능하다. memoir에는 이 패키지의 기능이 이미 포함되어 있으므로 별도로 로드하지 않아도 된다.

longtable과 tabularx를 합친 ltablex 패키지도 포함시켜야겠다. 이 패키지가 제공하는 `tabularx` 환경은 여러 페이지에 걸친 tabularx를 만들어준다.

표준 `tabular` 환경은 페이지를 넘길 수 없다. 몇 페이지에 걸치는 긴 표를 그려야 하는

경우도 드물지 않은데, 이 제한을 극복하기 위한 패키지들이 몇 개 있어서 여러 페이지에 걸치는 표를 그릴 수 있도록 해준다. longtable, supertabular, xtab 등이 그러하다.

table 안에서 색상을 쓰려면 colortbl 패키지가 필요하다. (역자) 이 패키지를 쓸 때 주의할 사항은 `\rowcolor` 명령이 반드시 그 행의 제일 처음 나오는 명령이어야 한다는 것이다. `\cellcolor` 명령이 더 효과적일 경우도 있다. 이밖에 `\columncolor` 명령도 제공한다.

xcolor 패키지의 `\rowcolors` 명령에 대해서는 6.6절을 참고하라.

Colour by row:

one	two	three
one	two	three
one	two	three

```
Colour by row:\\\vskip 2mm
\begin{tabular}{|l|c|r|}
  \hline
  \rowcolor{cyan}
  one & two & three\\
  \rowcolor{green}
  one & two & three\\
  \rowcolor{yellow}
  one & two & three\\
  \hline
\end{tabular}
```

깔끔한 트릭 하나. LaTeX으로 표를 그리는 것이 너무 복잡하다고 생각된다면 OpenOffice Calc와 Calc2LaTeX을 사용해보라. OpenOffice는 자유 스프레드시트 프로그램이고 Calc2LaTeX은 확장모듈인데 일정한 셀 범위를 LaTeX 테이블로 변환해준다. http://www.openoffice.org/, http://calc2latex.sourceforge.net/.

6.1 표의 행 간격

행의 높이는 그 안에 오는 텍스트의 높이에 따라 스스로 조절한다. 행 시작 **이전**에 간격을 추가하려면 특정 높이(height)와 0 길이를 가진 `\rule`로 시작하는 트릭을 쓸 수 있다. 행 **다음**에 간격을 추가하려면 `\\`에 추가 간격값을 선택 인자로 지정한다.

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

```
\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}
```

[역자 추가] 표 전체의 행간격을 일괄해서 바꾸려 할 때는 `\arraystretch` 값(기본값은 1)을 변경(`\renewcommand`)시켜준다. 단 이 값을 전역적으로 바꾸면 `tabular`만이 아니라 수식의 `array`도 영향을 받는다. 한 가지 더 추가하자면 setspace 패키지(memoir에는 이미 포함되어 있음)는 플로트 안의 내용(과 각주)을 행간격 1.0으로 조절하는 기능을 가지고 있다. 그러므로 `table` 안의 내용은 문서의 행간격과 별개로 기본 행간격 1.0으로 조판된다. 물론 `\arraystretch`로 이를 바꾸는 것이 가능하다.

6.2 괘선 두께

one	two	three
four	five	six

```
\begin{tabular}{|lll|}
\hline
%\setlength{\arrayrulewidth}{5pt}
one & two & three\\
\hline
four & five & six\\
%\setlength{\arrayrulewidth}{1pt}
\hline
\end{tabular}
```

[역자 추가] makecell을 이용하여 괘선 굵기를 바꾸어보는 예이다. makecell과 multirow 패키지가 필요하다.

First	Multicolumn head	
	Second multlined column head	Third column head
Cell text ... some and	A ... text more	28–31

```
\renewcommand\theadset{\def\arraystretch{.85}}%
\renewcommand\theadgape{}
{\begin{tabular}{!{\vrule width1.2pt}c
!{\vrule width1.2pt}c|c
!{\vrule width1.2pt}}
\Xhline{1.2pt}
\multirowthead{4}{First}&
\multicolumn{2}{c!{\vrule width1.2pt}}%
  {\thead{Multicolumn head}}\\
\Xcline{2-3}{.8pt}
& \thead{Second \\multlined \\ column head} &
\thead{Third \\ column head}\\
\Xhline{1.2pt}
Cell text & A &\multirowcell{4}{28--31}\\
... & ... &  \\
some & text & \\
and & more & \\
\Xhline{1.2pt}
\end{tabular}}
```

앞서 언급한 booktabs는 \toprule \midrule \bottomrule 명령을 제공하는데 이를 통하여 세로 괘선이 없는 표에서 가로 선의 굵기를 조절해준다.

6.3 숫자 정렬

테이블 안의 숫자들을 소수점 기준으로 정렬해야 하는 경우가 있다.

가장 간단한 방법은 @ 컬럼지시자를 이용하는 것인데 셀 안에 숫자만 있을 때는 쓸만하다. 컬럼분리자 &가 소수점으로 대치되게 하는 트릭이다.

3.14159
1.61803
1.41421
100.00000

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

다른 방법으로 dcolumn 패키지를 사용한다. 이 패키지는 D 컬럼지시자를 추가해주는데 세 개의 인자를 갖는다. LaTeX 소스와 출력에서 사용할 분리자(보통 둘 다 동일하게 '.'를 쓴다), 세 번째 것은 소수점 아래 표시할 자릿수. 세 번째 인자는 4.3과 같이 점 기준 왼쪽

과 오른쪽 자릿수를 지정할 수도 있다. 이 값이 −1이면 컬럼 내용은 분리자를 기준으로 가운데정렬된다.

이 컬럼의 모든 내용은 숫자라고 간주되어 수학 모드로 조판된다. 따라서 첫 행에 헤딩을 넣고자 한다면 \mbox 안에 텍스트를 두어야 한다.

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

```
\begin{tabular}{|D{.}{,}{4.2}|%
D{.}{.}{5}|D{.}{.}{-1}|}
\hline
\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 \\
\hline
\end{tabular}
```

6.4 slashbox 패키지

이 패키지는 \backslashbox 명령을 제공한다.

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

```
\begin{tabular}{|l|l|l|}
  \hline
  \backslashbox[2cm]{Lesson}{Date} &
  Monday & Tuesday \\
  \hline
  Stratigraphy & room A & room A \\
  Chemistry & room B & Lab $\alpha$ \\
  Physics & room C & Lab $\beta$ \\
  \hline
\end{tabular}
```

slashbox의 대각선은 여러 모로 품질이 만족스럽지 못하다. 근본적인 해결책은 아직 없는 것 같으나 makecell 패키지가 이와 유사한 기능을 제공하고 있기는 한데….

6.5 LaTeX 테이블로 데이터 가져오기

데이터 파일은 많은 사람에게 일용할 양식이다. 대부분의 데이터 파일은 숫자 컬럼을 가지고 있는 ASCII 텍스트이지만 어떤 사람들은 스프레드시트를 이용하기도 한다. 거의 모든 스프레드시트 프로그램은 시트를 ASCII 베이스의 .cvf 파일 포맷으로 내보내기할 수 있는데 이 파일은 ';' 문자를 분리자로 사용한다.

데이터 파일을 LaTeX 테이블로 변환하는 것은 지루한 작업이다. 아래의 Unix용 스크립트는 데이터 파일을 임의의 컬럼 수를 가진 테이블로 변환하는 것이다. .cvf 파일에 대해서도 동작한다.

```
#!/bin/sh

# dat2tex: converts tabular data to a tabular environment

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

# is this a cvs 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
```

6.6 그밖에 재미난 것

이 절은 역자가 임의로 추가한 것임

테이블에 관련해서는 재미있는 것이 많이 있다. 그만큼 문제점도 많은 셈이다. 여기서는 tabularcalc라는 것을 소개한다. 자세한 것은 패키지 문서를 읽으면 되고 샘플을 보이면 다음과 같다.

x	-4	-2	0	2.25	7
$2x-3$	-11	-7	-3	1.5	11
x^2	16	4	0	5.062,5	49
$\sqrt{x^2+1}$	4.123	2.236	1	2.462	7.071

```
\htablecalc[3]{$x$}{x=-4,-2,0,2.25,7}
  {$2x-3$}{2*x-3}
  {$x^2$}{x*x}
  {$\sqrt{x^2+1}$}{round(root(2,x*x+1),3)}
```

이 테이블에서 보다시피, x 값만 소스에 지정하면 결과는 TEX이 계산해주고 있다.

xcolor는 \rowcolors 명령을 제공한다. colortbl에도 비슷한 명령이 있지만 이 쪽이 조금 더 재미있다. \rowcolors를 쓰려면 xcolor에 [table] 옵션을 지정해야 한다.

test	row 1
test	row 2
test	row 3
test	row 4
test	row 5
test	row 6
test	row 7
test	row 8
test	row 9
test	row 10
test	row 11
test	row 12
test	row 13

```
\rowcolors[\hline]{3}{green!25}{yellow!50}\arrayrulecolor{red!75!gray}
\begin{tabular}{ll}
test & row \number\rownum\\ test & row \number\rownum\\
test & row \number\rownum\\ test & row \number\rownum\\
\arrayrulecolor{black}
test & row \number\rownum\\ test & row \number\rownum\\
\rowcolor{blue!25}
test & row \number\rownum\\ test & row \number\rownum\\
\hiderowcolors
test & row \number\rownum\\ test & row \number\rownum\\
\showrowcolors
test & row \number\rownum\\ test & row \number\rownum\\
\multicolumn{1}%
{>{\columncolor{red!12}}l}{test} & row \number\rownum\\
\end{tabular}
```

KTUG의 Faq 문서 'Tabular 환경'은 표 문제에 관한 거의 모든 해결책을 모아 둔 곳이다. 꼭 한 번 읽어볼 필요가 있다.

7 Tools 도구 메뉴

7.1 Tools/Mail Merges 도구/메일머지

이 유용하고 시간을 절약하게 해주는 도구는 textmerg 패키지로 구현된다. 어떤 문서가 있는데 거기에 표시되는 사람의 성과 이름, 호칭만을 바꾸어서 여러 장을 만들어야 하는 경우를 생각해보자. 그밖의 텍스트는 모두 똑같다.

세 개의 *field*를 정의한다. 각각 `\Name`, `\Surname`, `\Title`이라 하고 이것이 문서의 변하는 부분이 된다. 각각의 값은 외부 파일 `data.dat`로부터 가져오기로 하자.

```
\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 필드 `\Void`는 실제로 꼭 있어야 하는 것은 아니다. 그 앞에 마이너스 부호를 붙인 것은 데이터 파일에서 비어 있을 수 있다는 뜻이다. 이것을 둠으로써 레코드 사이를 빈 줄로 분리하려는 것이다.

`data.dat` 파일은 다음과 같이 작성한다.

```
Guido
Gonzato
Dr.

Francesco
Mulargia
Prof.

Marie
Curie
Mme
```

이것으로 준비는 끝났다. 출력되는 결과를 보면 불러온 텍스트가 포함되어 각 수신자마다 한 페이지씩 문서가 만들어진다.

7.2 Tools/Labels 도구/이름표만들기

메일 머지가 쉬웠다면 라벨 만들기는 훨씬 간단하다. 3×8 라벨용지에 20개의 라벨을 만들어야 하는 경우를 생각해보자. 사용할 패키지는, 짐작하겠지만 labels라는 것이다. 이 예에서 10개의 보통 라벨과 10개의 박스 라벨을 만들려고 한다.

```
\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{Linux system manager}}
  % now on to the boxed labels
  \boxedaddresslabel[\fboxsep=4mm\fboxrule=1mm]
  {\textbf{Guido Gonzato}, Ph.D.\\
  \textsl{Linux system manager}}
\end{document}
```

서로 다른 주소를 포함한 라벨을 만들려면 외부 파일을 이용할 수도 있고 메인 파일에 주소를 적어넣을 수도 있다.

```
\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 함께 쓰는 것도 가능하다. 한번 해보시라!

7.3 Tools/Default Language 도구/언어 설정

LaTeX의 기본 언어는 영어이다. 다른 언어도 지원한다. '언어 지원'이란 말은 예를 들면 'Chapter'나 'Index'와 같은 용어의 번역, 정확한 하이픈, 'ç'나 'é' 같은 문자를 키보드로

부터 직접 입력할 수 있는 기능 등을 의미하는 것이다. (이 글자들은 보통 `\c c`나 `\'e`와 같은 방식으로 입력한다.)

LaTeX 배포판에는 `language.dat`라는 파일이 들어 있다(보통 `$TEXMF/tex/generic/config/language.dat`). 이 파일에는 언어의 목록이 포함되어 있는데 이를 수정함으로써 원하는 언어의 하이픈 패턴을 선택할 수 있다.

XeTeX이나 LuaTeX과 같은 유니코드 텍 시스템이 도입됨으로써 언어 지원은 이전과 전혀 다른 양상이 되었다고 할 수 있다. XeTeX에서는 babel 대신 polyglossia 패키지를 써서 이전에 babel이 하던 역할을 대신한다. 한편 한국어 지원은 ko.TeX이 담당하고 있다.

만약 영어 사용자가 아니라면 babel 패키지를 다음과 같이 사용하면 된다.

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

☞ babel은 몇몇 문자의 동작을 해당 언어에 맞추어 변경시킨다. 글자가 사라지는 등 이상한 문제를 경험하면 나타나지 않는 글자를 `\charXX` 문법으로 삽입하면 될 것이다.

유럽어 입력에 관련된 이 부분은 우리같은 한글 사용자에게는 큰 문제거리가 아니다.

또한, 표준 ASCII 문자[3]가 아닌 부호붙은 글자들을 입력하려면 isolatin1 패키지가 포함되어야 한다. 그러나 이것은 권장하지 않는다. 왜냐하면 그것이 소스 파일의 가독성과 이식성을 감소시키기 때문이다. 계속 TeX 방식을 사용하는 편이 낫다.

한 글자를 입력하기 위해서 서너 번의 키조작을 해야 하는 것이 불만이라면 에디터 설정을 바꾸어서 이를 해결할 수 있을 것이다. 예를 들어 내가 쓰는 `jed`에서 'é'를 타이프하면 에디터가 이를 `\'e`로 바꾸어 입력하도록 설정해두었다. 이를 위한 나의 `.jedrc`이다.

```
define latex_mode_hook ()
{
  set_abbrev_mode (1);
  if ( () = abbrev_table_p ("LaTeX") )
    use_abbrev_table ("LaTeX");
#ifdef WIN32
  % prevent clash with movement keys
  undefinekey ("àà", "LaTeX-Mode");
  definekey (" \\`a", "àà", "LaTeX-Mode");
#else
  local_setkey (" \\`a",     "à");
#endif
  local_setkey (" \\'e",     "é");
  local_setkey (" \\`e",     "è");
  local_setkey (" \\`\\i{}", "ì");
  local_setkey (" \\`o",     "ò");
  local_setkey (" \\`u",     "ù");
}
```

자신의 에디터에 알맞은 방법은 에디터 도움말을 통해 찾아보라.

7.4 Tools/Hyphenation 도구/하이픈설정

LaTeX이 대체로 하이픈처리에 있어서 훌륭하지만 가끔 수작업으로 개입하는 것이 더 나은 결과를 가져올 때가 있다. 수동 하이픈은 `\-`를 단어 중간 하이픈이 필요한 곳에 적어넣으면 된다. 더 나은 방법으로 하이프네이션 규칙을 선언하는 방법도 있다.

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

[3] 컴퓨터 용어로서 '표준 ASCII 문자'라 함은 code 32(space)에서 126(tilde)까지의 글자들을 말한다.

위의 예에서 'earch'라는 단어는 하이픈처리하지 말도록 LaTeX에게 알려주고 있는 것이다. 특정 단어가 잘라지지 않게 하는 다른 방법으로 단어 전체를 \mbox 안에 넣어주는 방법도 있다.

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

7.5 Tools/Spell Check 도구/철자검사

LaTeX 자체는 철자검사와 아무 관련이 없다. 이것은 ispell, aspell 등 외부 프로그램을 이용해서 해야 한다. Unix에서 ispell을 다음과 같이 사용한다.

한국어 철자 검사는 명령행 인터페이스를 가진 것이 별로 없다. 아무튼 철자 검사가 에디터와 연동되는 것이 가장 바람직할 것이다. 다행히 오픈소스 한국어 철자검사기가 hunspell 사전으로 나와 있기 때문에 이를 이용하는 방법이 현재 개발되어 가고 있다. 예컨대 TeXworks 에디터는 hunspell 라이브러리를 채용하고 있으므로 부분적으로 한국어 철자검사가 가능해졌다.

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

-t 옵션은 ispell에게 TeX 및 LaTeX 명령을 무시하라고 알려주는 것이다. 주언어가 영여가 아니라면 적절한 사전을 -d 옵션으로 지정해주면 된다.

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

8 Help 도움말 메뉴

LaTeX에 대한 도움말을 얻는 데는 많은 방법이 있다. 온라인은 물론이고 오프라인 서적도 많다. 가장 좋은 출발점은 CTAN 사이트일 것이다. http://www.ctan.org/tex-archive/info/.

- info latex (Unix systems) 각종 명령, 개념에 대한 소략하지만 완전한 요약본을 보여주는 명령이다.
- http://www.giss.nasa.gov/latex/은 종합적인 온라인 참고문서이다. 유용한 링크가 많다.
- http://www.ctan.org/tex-archive/info/LatexHelpBook/은 LaTeX에 대한 훌륭한 도움말 시스템이다. Windows와 잘 통합되어 있다.
- news:comp.text.tex 뉴스그룹을 잊지 말자. 매우 유용한 도움말을 얻을 수 있다.

그리고 역자는 KTUG을 이 목록에 추가하지 않을 수 없다.

2010년 현재 대부분의 GNU/Linux 배포판은 가장 완전한 TeX/LaTeX 시스템일 TeX Live를 포함하고 있다. 수많은 안내문서가 제공된다. 나의 Ubuntu 기계에서 그것은 /usr/share/doc/texlive-doc/ 에서 찾을 수 있다.

texdoc 명령행 인터페이스에 익숙해지면 원하는 대부분의 문서를 즉시 읽을 수 있게 될 것이다.

9 마지막으로

 이 가이드가 유용하기를 진심으로 바란다. 제안과 비평이 있다면 저자에게 연락해주기 바란다.

번역본의 라이센스도 동일하다. 번역에 관련된 문제는 역자에게 연락해주기 바란다.

A 문서 본보기

`article` 클래스에 대한 본보기 문서는 2.1절에 제시되어 있다. 이어지는 샘플들은 더 많은 예제들이다.

```
\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{type1cm}
\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}

\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.ps}
  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.

다음 본보기는 한글 문서용으로서 역자가 추가한다.

```
\documentclass[a4paper,footnote]{oblivoir}

\usepackage{fapapersize}
\usefapapersize{*,*,20mm,60mm,30mm,*}

\usepackage[svgnames]{xcolor}
\usepackage{multirow}
\usepackage{iftex}
\ifXeTeX
 \defaultfontfeatures{Mapping=tex-text}
 \setmainfont{Times New Roman}
 \setkormainfont{나눔명조}
\else\ifPDFTeX
 %... pdftex stuffs
\fi
\fi

\hypersetup{colorlinks,urlcolor=blue,filecolor=magenta}

\begin{document}
\title{본보기 문서}
\author{저자명}

\maketitle

\section{절 제목}

본문

\end{document}
```

그림 A.6: 한글 문서 샘플