

HzGuide

테크니컬 라이터를 위한 레이텍 클래스



사용 설명서
2022 | 2.0

머리말	7
1 개요	9
1.1 클래스 옵션	9
1.2 공통 설정 파일	10
1.3 다국어 문서	11
2 페이지 모양과 문서 구조	13
2.1 판형	13
2.2 면주	15
2.3 장, 절, 문단	17
2.4 차례와 북마크	19
2.5 장 차례	19
2.6 절 앞에서 쪽 나누기	20
2.7 사용자 행동	20
2.8 워터마크	21
2.9 반달 색인	21
3 그림과 표	25
3.1 문단 사이에 이미지	25
번역사를 위한 이미지 대조	28
3.2 설명 딸린 이미지	29
3.3 글줄 안에 이미지	32
3.4 가짜 이미지	32
3.5 캡션	33
3.6 콜아웃	33
3.7 이미지 파일 이름과 번호	35
3.8 앨범 만들기	35
3.9 표	35
제품 사양을 보여주는 표	37
제품 구성품을 보여주는 표	38
3.10 표 안에 이미지	38
4 목록과 용어	41
4.1 나열과 절차	41
4.2 정의 목록	42
4.3 선택 목록	44
4.4 용어	45
4.5 첨자	46
4.6 색인	47
4.7 원 숫자	48

4.8 기호	49
5 글상자와 경고문	51
5.1 글상자	51
5.2 경고문 표어	54
6 문서 제어 도구	55
6.1 개체	55
6.2 상호 참조와 하이퍼링크	56
6.3 확정되지 않은 사항들	56
6.4 TSV 또는 CSV 파일 읽기	57
6.5 싱글 소스 퍼블리싱	58
조건식	59
이미지 경로	60
텍 파일 경로	62
파일 외부에서 설정 변경하기	63
7 소스 코드	65
7.1 레이텍 명령	65
7.2 코드 예시하기	65
7.3 공백과 빈 줄이 유지되는 환경	67
8 문서 표지	69
9 문서 템플릿	73
A 파이선 스크립트	75
A.1 ltx.py: 텍 파일 컴파일하기	76
A.2 i.py: 더 게으른 사용자를 위한 레이텍 래퍼	77
A.3 iu.py: 이미지를 다른 포맷으로 변환하기	79
A.4 fontinfo.py: 폰트 정보 보기	80
A.5 tlconf.py: 텍 라이브와 관련된 설정	80
A.6 op.py: 파일 열기	83
A.7 wordig.py: 문자열을 찾아 바꾸기	84
A.8 unit.py: 단위 변환하기	86
A.9 fu.py: 파일들 백업하기	87
파일들 백업하기	87
파일 이름 바꾸기	88
파일들 모으기	89
전체 파일 크기 구하기	90
A.10 trapper.py: 스크린샷 저장하기	90
A.11 레이텍 템플릿 모음	90
A.12 이따금 요긴한 스크립트들	93

B 응용	95
찾아보기	99

그림 차례

2.1 이 문서의 레이아웃	15
3.1 초고 이미지	26
A.1 multilingual.pdf	91

머리말

<https://github.com/YiHoze/HzGuide>

나는 테크니컬 라이터로서 내게 필요한 것들을 HzGuide 클래스에 만들어 넣었다. memoir 클래스와 expl3 문법에 친숙한 사용자는, 그런 사용자가 HzGuide 클래스를 사용하지 않겠지만, 이 문서를 이해하는 데에 어려움이 없을 것이다.

텍스트 에디터로 HTML 파일을 읽는 것은 불가능하지 않을지라도 매우 피곤한 일이다. 텍 문서가 그와 같이 복잡하다면 글을 쓰는 것은 물론이고 고치는 것도 고역이다. 모든 레이텍 사용자들이 지향하듯이 나 역시 “판독하기 쉬운 텍 문서”를 만드는 것을 목표로 매크로들을 만들었다.

```
\begin{figure}  
\frame{\includegraphics[scale=1.25]{foo.jpg}}  
\caption[그림 차례에 들어가는 짧은 설명]{긴 설명}  
\label{foo}  
\end{figure}
```

이러하면 아래 코드로 위의 것과 동일한 결과를 얻을 수 있다.

```
\image*[scale=1.25]{foo}(긴 설명)<짧은 설명>
```

일반적으로 설명서에서 그림과 차례에 캡션을 붙이지 않는다. 다른 종류의 문서들에 비해 설명서가 더 많은 이미지들을 포함하기 때문에 떠다니는 figure 환경이 사용하기에 도리어 부적합하다. 하지만 그런 것들이 필요한 경우에 대응할 수 있도록 \image 명령이 설계되었다.

그 다음 목표는 “제어하기 쉬운 문서”이다. \image 명령의 짝이 \ImageSetup이다. HzGuide 클래스에 정의된 많은 명령들이 그와 같은 쌍으로 이루어져 있다. 그 설정 명령들을 이용하여 문서 전체에 걸쳐 이미지 크기를 비롯한 스타일들을 손쉽게 변경할 수 있다.

그와 같은 목표 아래 매크로를 생성하는 매크로들이 있다. \NewTerm, \NewTerms, \NewBoundedBox 그리고 \NewConditionals가 명령 또는 환경들을 만드는 명령들이다.

\NewTerm 명령의 목적 중 하나가 번역에 있다. 한 문서에 포함된 텍스트는 번역의 관점에서 세 가지로 나뉜다. 번역되어야 할 텍스트, 번역되지 않아야 할 텍스트, 이미 정해진 것으로 바뀌어야 할 텍스트. 리모컨 버튼들이 번역되지 않아야 할 텍스트이다. 스타크래프트 I의 한국어 버전에서 “marine”이 “마린”이었는데, 스타크래프트 II에서는 “해병”으로 바뀌었다. 번역자는 사용자 인터페이스에 사용되는 것들을, 설명 그것이 부적절할지라도, 따라야 한다. 번역자가 번역해야 할 텍스트와 하지 않아야 할 텍스트를

식별할 수 있도록 다음과 같은 꼬리표가 필요하다.

```
\invariable{Volume} \predefined{Marine}
```

전자 문서가 보편화되면서 진부해진 것들이 있다. 장 chapter 이 오른 페이지에서 시작되는 것과 같은 일반적인 조판 관행에 텍 사용자들은 시나브로 익숙해지지만, 인쇄를 전제로 하지 않는 문서들이 점점 많아지면서 빈 페이지를 문서의 자연스러운 일부로 받아들이지 않는 독자들이 늘어나고 있다. PDF 뷰어의 북마크와 검색 기능 때문에 면주와 색인은 더 이상 독자들에게 긴요한 장치로 간주되지 않는다.

같은 이유로, 내가 처음 만들 때 의도했던 것과는 달리, HzGuide 클래스에 정의된 여러 매크로들 중 일부는 이제 장식에 가까운 것이 되었다. 2.1 절에서 소개하는 변이단 조판에 내가 집착했던 이유는 가장 이상적인 글줄 길이인 12 센티미터를 고수하기에 A4 종이가 너무 컸기 때문이다. 하지만 이제 종이 크기나 글자 크기는, 적어도 대부분의 나의 의뢰인들에게는, 그다지 문젯거리가 아니다. 독자들은 저마다 편한 크기로 페이지를 확대하거나 축소해서 볼 수 있다. 장 차례와 워터마크와 반달 색인이 비슷한 이유로 거의 쓸모를 잃었다.

나는 실제 프로젝트에서 더 이상 변이단을 사용하지 않는다. 하지만 HzGuide로 할 수 있는 것들을 가급적 많이 보여주기 위하여 이 문서에 변이단을 사용하였다. 그리고 색인도 추가하였다. 색인이 예외적으로 필요한 문서가 바로, *The L^AT_EX3 Interfaces*^{expl3} 처럼, 많은 매크로를 포함하는 레이텍 문서이다.

HzGuide 클래스는 expl3로 작성되었다. 처음부터 클래스 작성을 목표로 한 것은 아니었다. 잡다한 서너 패키지들을 만들었고, 정확히 언제인지 기억하지 못하지만 그것들을 합쳐서 클래스로 만들었다. 그리고 2015 년에 expl3를 이용하여 새로 작성하였다. expl3가 아니었다면 매크로를 생성하는 매크로들을 만들 엄두를 내지 못했을 것이다.

이 호재

제 1 장

개요

HzGuide는 memoir에 기반하여 테크니컬 라이터 technical writer의 설명서 작성을 돕기 위한 목적으로 만들어진 레이텍 클래스이다.

1.1 클래스 옵션

memoir 클래스 옵션들과 함께 다음 옵션들을 사용할 수 있다.

```
\documentclass[language=japanese, styleset=../foo.tex, Noto]{hzguide}
```

language = korean, english, chinese, TC, czech, danish, dutch, finnish, german, italian, japanese, norwegian, polish, portugeue, russian, slovakian, spanish, swedish, turkish

영어는 language=english 대신 english로 지정할 수 있다. 디폴트는 korean이다. chinese는 간체 简体, TC는 번체 繁體이다. 언어에 따라 polyglossia 또는 xeCJK 패키지가 호출된다.

property = foo, ...

주어진 문서 속성들이 \DocumentSetup에 전달된다. 59 페이지 [조건식](#)을 보라.

styleset = foo.tex

지정된 공통 설정 파일을 불러들인다. 문서들 사이에 일관성을 유지하려면 전제부 preamble에서 선언되는 설정들을 하나의 파일에 담아 여러 문서에 공통으로 사용해야 한다. 이 옵션의 실재는 \input 명령의 실행에 불과하나, 설정 파일을 클래스 옵션으로서 명시함으로써 문서의 정체성을 확인하는데에 이것의 목적이 있다.

packageset = packages.tex

특정한 순서로 패키지들을 가져와야 하는 예외적인 경우를 위해 이 옵션이 고안되었다. 예를 들어 전제부에 다음과 같이 선언하면 HzGuide에 의해 kotex이 이미 올라온 뒤에 bidi가 읽힌다. 그것은 공백 없이 영어 텍스트와

이어진 한글이 식자되지 않는 문제를 일으킨다. 다음과 같이 packages.tex 을 작성하고 packageset 옵션에 지정하면, 이 패키지들이 kotex보다 먼저 올라온다.

```
\usepackage{stackengine}
\usepackage{ulem}
\usepackage{bidi}
```

Noto = true/false

Noto 폰트가 사용된다.

minted = true/false

minted 패키지가 사용된다. \coderead 명령이 \verbatiminput 대신 \inputminted를 사용하게 된다.

pairquote = true/false

csquotes 패키지가 사용된다.

template = true/false

hztemplate.tex을 불러들인다. 설명서 초안을 만들기 위한 목적으로, 설명서에 사용되는 여러 요소들을 예시하는 명령들을 제공한다. 73 페이지 [문서 템플릿](#)을 보라.

1.2 공통 설정 파일

공통 설정 파일을 작성할 때, 폰트나 다른 이유로, 텍 엔진이나 문서의 언어에 따라 설정을 달리해야 하는 경우가 있다. 다음 명령들이 그런 경우에 도움이 될 것이다.

```
\IfXetex{ TRUE }[ FALSE ]
\IfLuatex{ TRUE }[ FALSE ]
\ifLang{언어}{ TRUE }[ FALSE ]
\begin{IfLanguage}{언어} \end{IfLanguage}
```

\IfXetex은 현재 문서가 지텍 XeTeX 에 의해 컴파일될 때, \IfLuatex은 루아텍 LuaTeX 이 사용될 때 유효하다.

```
\documentclass[language=german]{hzguide}
\ifLang{german}{ ... }
\begin{IfLanguage}{german} ... \end{IfLanguage}
```

\ifLang 명령과 IfLanguage 환경은 \NewConditionals 명령에 의해 만들어졌다. 이 때 크로들의 보다 자세한 사용법에 대해 59 페이지 [조건식](#)을 보라.

두 가지 이상의 문서에 공용으로 사용해야 하는 부가적인 설정 파일을 만들어야 하는 경우에, 그 파일을 서로 다른 폴더에서 읽어야 한다면, 다음과 같은 명령이 유용할 수 있다.

```
\NewDocumentCommand \styleinput { m }
{
  \IfFileExists{ ../#1 }
```

```

{
  \input{../#1}
}{
  \IfFileExists{ #1 }{ \input{#1} }{}
}
\styleinput{common.tex}

```

이 명령은 지정된 파일을 상위 폴더에서 찾고 없으면 현재 폴더에서 찾아서 가져올 것이다.

1.3 다국어 문서

인쇄나 다른 이유로 여러 언어로 번역된 PDF 문서들을 하나로 합쳐야 할 때, 각 문서에 그 문서의 언어를 나타내는 북마크를 추가하는 것이 유용할 것이다.

```

\NewDocumentCommand \BookmarkLanguage { m 0{-1} }
{
  \bookmark[level=#2, page=1]{#1}
}
\BookmarkLanguage{ENGLISH}

```


제 2 장

페이지 모양과 문서 구조

2.1 판형	13
2.2 면주	15
2.3 장, 절, 문단	17
2.4 차례와 북마크	19
2.5 장 차례	19
2.6 절 앞에서 쪽 나누기	20
2.7 사용자 행동	20
2.8 워터마크	21
2.9 반달 색인	21

2.1 판형

\LayoutSetup 명령을 이용하여 판면과 여백을 설정할 수 있다.

```
\LayoutSetup{
  paper=A4,
  column=vartwo
}
```

paper = A3, A4, A5, A5V, letter, B5, JB5, slide, arbitrary

종이 크기들은 다음과 같다.

A3 297 × 420 mm

A4 210 × 297 mm

A5 148 × 210 mm

A5V 152 × 225 mm

이것은 신국판이다.

letter 8.5 × 11 in (215.9 × 279.4 mm)

B5 176 × 250 mm

JB5 182 × 257 mm
 이것은 JIS의 B5 크기이며 흔히 사륙배판이라고 불린다.
 slide 9 × 6 in (228.6 × 152.4 mm)

일반적인 규격에서 벗어나는 판형을 만들려면 `arbitrary`를 지정하고, 아래 옵션들을 적절하게 설정하라.

`stockwidth` = 250mm
 인쇄 용지의 폭. 재단선이나 반달 색인을 표시해야 할 때 페이지보다 넓은 크기가 지정되어야 한다.

`stockheight` = 353mm
 인쇄 용지의 높이

`paperwidth` = 210mm
 페이지 폭

`paperheight` = 297mm
 페이지 크기

`landscape` = true/false
 페이지가 가로 방향으로 바뀐다.

`column` = one, vartwo
 vartwo를 지정하면, 이 문서와 같이, 변이단 變二段으로 조판된다.

`ulmargin` = 36mm
 상단 여백

`ulratio` = 1.0
 하단 여백을 결정할 배율을 지정하라. “2”를 지정하면 하단 여백이 상단 여백의 두 배가 된다.

`lrmagin` = 35mm
 좌우 여백

`vartwomargin` = 20mm
 변이단으로 조판할 때 오른쪽 여백. 왼쪽 여백은 오른쪽 여백의 세 배가 된다.

`showtrims` = true/false
 재단선이 표시된다. 그림 2.1을 보라.

`showlayout` = true/false
 이 옵션은 `\ShowPageLayout` 명령을 호출하여 여백 영역과 텍스트 영역의 테두리에 선을 그린다. 그림 2.1을 보라.

`hook` = `\setheadfoot{0mm}{5mm}`
 이 예와 같은 구획 명령을 이 설정에 끼워 넣을 수 있다.

변이단에서 사용할 수 있는 `IfVartwoEnlarge` 환경은 주어진 길이 만큼 글줄의 폭을 왼쪽 여백으로 확장한다. 디폴트는 `\marginparwidth`와 `\marginparsep`의 합이다.

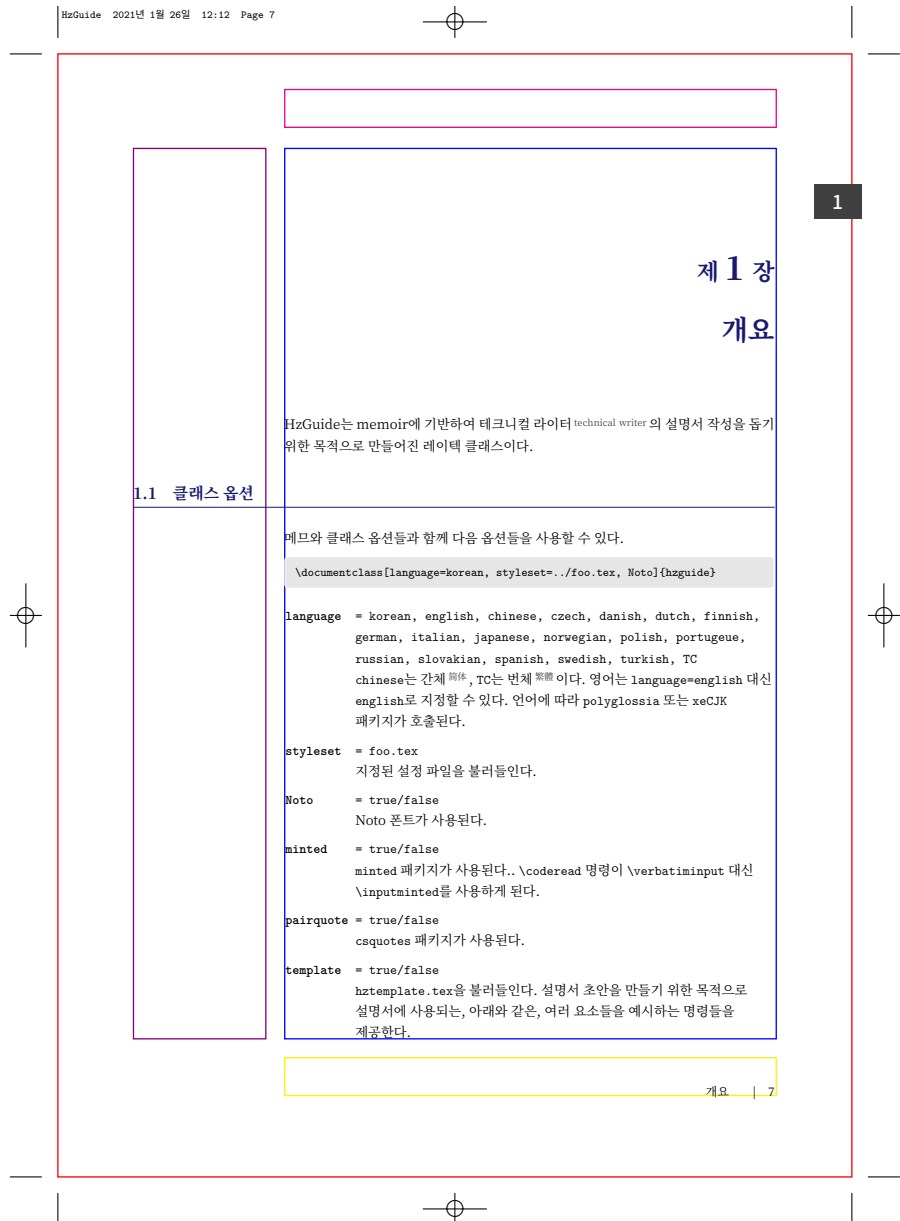


그림 2.1: 이 문서의 레이아웃

```
\begin{IfVartwoEnlarge}[폭] \end{IfVartwoEnlarge}
\IfVartwo{ TRUE }[ FALSE ]
```

\IfVartwo 명령은 클래스 내부에서 여러 명령들의 작동을 제어한다.

2.2 면주

\PageStyleSetup 명령을 이용하여 면주 모양을 변경할 수 있다. \LayoutSetup 명령이 이

명령을 이용한다.

```
\PageStyleSetup{
  headrule=true,
  evenheadleft=\leftmark,
  oddheadright=\rightmark,
  evenfootleft=\thepage,
  oddfootright=\thepage,
  chaptermark=true,
  sectionmark=true,
}
```

font = \rmfamily\small
 폰트

headleft = \leftmark, \rightmark, \thepage, 텍스트
 상단 왼쪽

headcenter 상단 가운데

headright 상단 오른쪽

headinner 상단 안쪽 (왼쪽 페이지의 오른쪽과 오른쪽 페이지의 왼쪽)

headouter 상단 바깥쪽 (왼쪽 페이지의 왼쪽과 오른쪽 페이지의 오른쪽)

footleft 하단 왼쪽

footcenter 하단 가운데

footright 하단 오른쪽

footinner 하단 안쪽

footouter 하단 바깥쪽

evenheadleft 왼쪽 페이지의 상단 왼쪽

evenheadcenter 왼쪽 페이지의 상단 가운데

evenheadright 왼쪽 페이지의 상단 오른쪽

oddheadleft 오른쪽 페이지의 상단 왼쪽

oddheadcenter 오른쪽 페이지의 상단 가운데

oddheadright 오른쪽 페이지의 상단 오른쪽

evenfootleft 왼쪽 페이지의 하단 왼쪽

evenfootcenter 왼쪽 페이지의 하단 가운데

evenfootright 왼쪽 페이지의 하단 오른쪽

oddfootleft 오른쪽 페이지의 하단 왼쪽

oddfootcenter 오른쪽 페이지의 하단 가운데

oddfootright 오른쪽 페이지의 하단 오른쪽

headrule = true/false
상단 가로선

footrule = true/false
하단 가로선

chaptermark = true/false
\chaptermark 명령이 활성화된다.

sectionmark = true/false
\sectionmark 명령이 활성화된다.

markdelimter = \quad
주어진 문자가 장절 번호와 제목 사이에 삽입된다.

chapterpage = true/false
상단 면주가 장 페이지에도 적용된다.

2.3 장, 절, 문단

\HeadingSetup 명령을 이용하여 장과 절 제목의 모양을 그리고 문단 모양을 변경할 수 있다.

```
\HeadingSetup{
  linespacing=1.25,
  chapterfont=\normalfont\bfseries,
  sectionsize=\Large\color{MidnightBlue},
  paragraphstyle=indent
}
```

linespacing = 1.25

줄 간격. 이것은 \baselinestretch를 설정하는 것과 같다.

article = true/false

면주가 단순해지고 차례 표제가 절 수준으로 바뀐다. 이것은 memoir의 article 클래스 옵션과 무관하다.

cftsection = true/false

차례 표제가 절 수준으로 바뀐다.

chapterwider = true/false

변이단 조판에서 장 제목이 왼쪽 여백에서 시작되게 한다.

chapterstyle = tight, sparse

장 스타일. 이것은 \chapterstyle를 설정하는 것과 같다. 그러나 memoir 클래스에 정의된 여러 장 스타일들 가운데 일부에 대해서만 아래 옵션들이 유효할 것이다. tight 스타일은 디폴트에서 줄 간격을 줄인 것에 불과하다. veelo 같은 스타일을 이용하려면, \printerchaptername을 비롯한 여러 매크로들을 재정의해야 한다.

`cjkchapter = true/false`

한국어에 대해서는 “제 1 장” 이, 중국어와 일본어에 대해서는 “第 1 章” 이 식자된다.

`chapteralign = \raggedleft, \centering, \raggedright`

장 제목의 정렬

`chapterfont = \normalfont\bfseries`

장 제목의 폰트

`chaptercolor = black`

장 제목의 색

`chapternamesize = \huge`

장 이름의 크기

`chapternumbersize = \HUGE`

장 번호의 크기

`chaptertitlesize = \Huge`

장 제목의 크기

`nochaptername = true/false`

장 이름이 식자되지 않는다. “제” 도 식자되지 않는다.

`chaptercontents = true/false`

장 제목 아래에 장 차례가 만들어진다.

`sectionstyle = firm, sparse, tight, multiline, rule`

절 스타일. `multiline`은 변이단 조판에서 절 제목의 둘째 줄이 들여쓰기 되는 것을 방지한다. 두 가지 옵션을 함께, 이를테면 `multiline`과 `rule`을 동시에 지정할 수 있다.

`sectionalign = \raggedleft, \centering, \raggedright`

절 제목의 정렬

`sectionfont = \normalfont\bfseries`

절 제목의 폰트

`sectionsize = \Large`

절 제목의 크기

`subsectionsize = \large`

하위 절 제목의 크기

`subsubsectionsize = \normalsize`

최하위 절 제목의 크기

`paragraphstyle = interval/indent`

절 모양. 디폴트는 `interval`이다. 이것은 들여쓰기 없이 문단 사이를 띄운다. 설정에 따라 목록들의 항목 간격도 함께 조정된다.

`parskip = 0.25\onelineskip`

문단 간격. `paragraphstyle` 옵션에 `interval`을 지정했을 때 이 값이 유효하다.

`parindent = 1em`

들여쓰기 크기. `paragraphstyle` 옵션에 `indent`를 지정했을 때 이 값이 유효하다.

2.4 차례와 북마크

“차례”가 차례에 포함되어 있는 것은 군더더기처럼 보인다. 별표가 붙은 `\tableofcontents*`는 “차례”를 차례에 넣지 않지만, 북마크에도 차례가 추가되지 않는다.

```
\TableOfContents[줄 간격]
\TableOfContents*[1.1]
\ListOfFigures[줄 간격]
```

`\TableOfContents` 명령은 “차례”를 차례에 넣지 않는 대신 북마크에 차례를 추가한다. 이 명령에 줄 간격을 선택적으로 지정할 수 있다. 별표가 붙은 `\TableOfContents*`는 `\tableofcontents`와 동일하다. `\ListOfFigures` 명령을 같은 방법으로 사용할 수 있다.

2.5 장 차례

설명서에서 장 차례가 필요한 부분이 “문제 해결하기” `troubleshooting`이다. `\HeadingSetup`에 `chaptercontents` 옵션을 지정하면 `\ChapterContentsEnable` 명령이 호출된다.

```
\ChapterContentsEnable[장 번호](절 수준)
\ChapterContentsDisable
```

장 번호 없이 `\ChapterContentsEnable`을 선언하면 모든 장에 장 차례가 만들어진다. 장 번호를 지정하면 그 장에만 차례가 만들어진다. 절 수준에 “3”을 지정하면 최하위 절 `\subsubsection`까지 장 차례에 포함된다. 디폴트는 “2” `\subsection`이다. 아래 예와 같이 `\ChapterContentsDisable` 명령을 함께 사용하면 장 차례가 만들어져야 할 장의 범위를 지정할 수 있다.

```
\ChapterContentsEnable
\chapter{troubleshooting}
...
\chapter{Frequently Asked Questions}
...
\ChapterContentsDisable
\chapter{Warranty}
```

2.6 절 앞에서 쪽 나누기

모든 절이 새 페이지에서 시작하는 것은 대개 바람직하지 않지만, 일정한 양식이 반복되는 참조 설명서 `reference manual`에서는 독자에게 오히려 도움이 될 수 있다.

```
\SectionNewpageOn*
\SectionNewpageOff
\SubsectionNewpageOn
\SubsectionNewpageOff
```

`\SectionNewpageOn` 명령이 주어지면 `\SectionNewpageOff` 명령을 만날 때까지 각 장에서 둘째 절 이후의 모든 절 앞에서 쪽 나눔이 일어난다. `\SectionNewpageOn*` 명령은 첫째 절조차 다음 페이지로 넘긴다. `\SubsectionNewpageOn`과 `\SubsectionNewpageOff` 명령들도 같은 방식으로 작동한다.

2.7 사용자 행동

하나의 사용자 업무 `user task`는 하나 이상의 사용자 행동이 연결되어 달성된다. 소프트웨어 제품에서, 특히 모바일 앱에서, 매우 다양한 기능들이 일련의 단순한 조작으로 작동한다. 아래 예와 같은 설명은 “목적”과 “방법” 또는 “심리적 행동”과 “물리적 행동”으로 이루어져 있다.

Turn iPhone on. Press and hold the Sleep/Wake button until the Apple logo appears.

Unlock iPhone. Press either the Sleep/Wake or Home button, then drag the slider.

이와 같은 단락을 만드는 데에 `\paragraph` 명령이 충분하지만, 자유롭게 스타일을 변경할 수 있도록 `\action` 명령이 고안되었다.

```
\ActionSetup{옵션}
\action{목적 텍스트}
```

`\ActionSetup` 명령으로 다음 옵션들을 변경할 수 있다.

```
beforeskip = 0.5\hzbarskip
            단락 위 간격

afterskip  = -\hzbarskip
            단락 아래 간격. inline 옵션이 true이면 이 간격이 무시된다.

font       = \bfseries
            폰트

delimiter  = :, \quad
            구획 문자.

inline     = true/false
            이 옵션이 false이면 방법 설명이 다음 줄에서 시작한다.
```

이 단락들은 외형적으로 목록과 흡사하다. 하지만 이것들은, 각 단락의 화제^{topic}가 서로 다르기 때문에, 목록이 될 수 없다.

2.8 워터마크

\watermark 명령을 이용하여 워터마크를 삽입할 수 있다.

```
\watermark{
  x=15, y=150,
  fontsize=45, color=yellow, rotate=-90,
  text={WATERMARK}
}
```

워터마크를 없애려면 \watermark가 선언된 페이지로부터 적어도 1 페이지 뒤에서 \ClearWatermark 명령이 주어져야 한다.

```
\WatermarkSetup{옵션}
\watermark{옵션}
\ClearWatermark
```

\WatermarkSetup 명령으로 변경할 수 있는 옵션들은 다음과 같다.

x	= 10	x 좌표. 단위는 밀리미터이다.
y	= 10	y 좌표
color	= blue	색
font	= \sffamily	폰트
fontsize	= 45	폰트 크기. 단위는 포인트이다.
rotate	= 90	기울기. 1 사분면에서 반시계 방향으로 주어진 각도만큼 회전한다.
text	= 텍스트	워터마크 텍스트
image	= 이미지 파일	text 옵션과 image 옵션이 동시에 설정된다면, 앞서 주어진 옵션이 무시된다.
scale	= 1.0	주어진 배율만큼 이미지가 확대되거나 축소된다.

WATERMARK

2.9 반달 색인

\ThumbIndexEnable 명령을 이용하여 장 번호나 제목을 반달 색인^{thumb index}에 넣을 수 있다. 이 명령은 반달 색인을 오른 페이지에 만든다.

```
\ThumbIndexSetup{옵션}
\ThumbIndexEnable
\ThumbIndexDisable
```

이 기능은 \chaptermark 명령을 이용하기 때문에 \chapter 명령이 주어지지 않으면 발효되지 않는다. 이 문서는 다음과 같이 설정되었다.

```
38 \ThumbIndexSetup{
39     xoffset=0mm, toffset=50mm, boffset=50mm,
40     width=3em, height=2em, interval=5mm,
41     fgcolor=white, bgcolor=darkgray,
42     style=\sffamily\bfseries\LARGE\centering,
43     content=\thechapter
44 }
45 \makeatletter
46 \patchcommand{\@smemmain}{\ThumbIndexEnable} % \mainmatter*
47 \makeatother
48 \patchcommand{\printindex}{\ThumbIndexDisable\cleartoverso}{}
```

\ThumbIndexSetup 명령을 이용하여 반달 색인의 스타일을 바꿀 수 있다. 위치나 크기 옵션에 밀리미터 단위로 값을 지정하는 것이 좋다. 주어진 값들로부터 밀리미터 단위로 변환하여 좌표를 구하기 때문에 그리고 이 과정에서 소수점 이하가 버려지기 때문에 정확한 계산을 기대하기 어렵다.

xoffset	= 0mm	반달 색인이 바깥쪽으로 돌출되는 크기. 그림 2.1을 보라.
toffset	= 30mm	첫 장에서 반달 색인의 y 좌표. 엄밀히 말해 이것은 페이지 상단으로부터 피어야 하는 최소 거리이다.
boffset	= 30mm	페이지 하단으로부터 피어야 하는 최소 거리
interval	= 5mm	앞 장의 반달 색인과 뒷 장의 반달 색인 사이의 간격
width	= 3em	박스 폭
height	= 2em	박스 높이
fgcolor	= white	글자 색
bgcolor	= gray	박스 색

style = `\bfseries\centering`
 폰트와 정렬
vertical = `true/false`
 박스가 세로 방향으로 회전한다.
content = `\thechapter`
 장 제목을 표시하는 방법에 대해 아래 예를 보라.

```

\ThumbIndexSetup{
  xoffset=12.1em, toffset=30mm, boffset=60mm,
  width=15em, height=2.5em, interval=5mm,
  fgcolor=white, bgcolor=darkgray,
  style=\sffamily\bfseries\Large,
  content=\quad\leftmark,
  vertical=true
}

```


제 3 장

그림과 표

3

3.1 문단 사이에 이미지	25
번역사를 위한 이미지 대조	28
3.2 설명 딸린 이미지	29
3.3 글줄 안에 이미지	32
3.4 가짜 이미지	32
3.5 캡션	33
3.6 콜아웃	33
3.7 이미지 파일 이름과 번호	35
3.8 앨범 만들기	35
3.9 표	35
제품 사양을 보여주는 표	37
제품 구성품을 보여주는 표	38
3.10 표 안에 이미지	38

다른 종류의 문서들에 비해 설명서가 갖는 두드러진 특징들 중 하나는 그림과 표를 많이 포함한다는 것이고 그래서 그것들의 배치를 텍에 맡겨 놓기 곤란하다는 것이다.

3.1 문단 사이에 이미지

\image 명령이 그림을 문단 사이에 둔다.

```
\image[scale=0.5]{uncertain}
```

(초고에서 그림 자리를 보여주기 위한 이미지)<초고 이미지>



그림 3.1: 초고에서 그림 자리를 보여주기 위한 이미지

```
\image*!{옵션}[이미지 파일][다른 이미지 파일](캡션)<짧은 캡션>
\ImageSetup{옵션}
```

일반 출판물과 달리 설명서에서 그림은 보충적인 자료가 아니라 핵심적인 정보의 일부이기 때문에 캡션이 오히려 군더더기이다. 불가피한 경우에만 그림에 캡션을 붙여야 한다.

`beforekip` = `\hnullldim, 0.5\onelineskip`

그림 위 간격. 값이 `\hnullldim`이면 `\ObjectSetup`으로 설정된 값이 사용된다. 55 페이지 [개체](#)를 보라.

`afterkip` = `\hnullldim, 0.5\onelineskip`

그림 아래 간격.

`gap` = `1em`

다른 이미지 파일이 주어졌을 때 두 이미지 사이 간격

`align` = `\raggedleft, \centering, \raggedright`

이미지 정렬

`float` = `true/false`

이 옵션은 이미지를 `figure` 환경에 넣는다. 다시 말해 그림의 위치가 텍에 의해 결정된다.

`float-placement` = `tbh`

`figure` 환경을 이용하는 경우에 위치 옵션

`frame` = `true/false`

이미지 테두리에 선이 그어진다.

`fitline` = `true/false`

이미지가 `fitwidth`에 설정된 폭보다 크면 그에 맞게 줄어든다.

`fitwidth` = `\linewidth`

이미지 폭

`flush` = `true/false`

변이단 조판에서 이미지가 왼쪽 여백으로 넘어간다.

`scale` = `1.0`

주어진 배율만큼 이미지가 확대되거나 축소된다.

`landscape` = true/false
이미지가 90도 회전한다. 캡션도 함께 회전한다.

`middle` = true/false
이미지가 주어진 공간에서 세로로 중간에 놓인다.

`caption` = 텍스트
캡션. 파일 이름 따위를 나타내기 위한 목적으로 밑줄 문자(_)를 사용할 수 있다.

`caption-short` = 텍스트
그림 차례에 들어가는 짧은 캡션

`caption-verb` = true/false
기호 문자들을 백슬래시 없이 그대로 *verbatim* 캡션에 사용할 수 있다.

`caption-ignore` = true/false
캡션이 식자되지 않는다.

`caption-fit` = true/false
캡션의 폭이 이미지 폭에 맞춰진다.

`label` = 텍스트
상호참조를 위한 라벨. 달리 지정하지 않으면 이미지 파일의 이름이 라벨이 된다.

`legend` = true/false
캡션에 번호가 붙지 않는다.

`showfilename` = true/false
이미지 파일의 이름이 표시된다.

`fake` = true/false
32 페이지 **가짜 이미지**를 보라.

`star` = 옵션
이것은 기본 설정과 다른 작동을 허용하기 위한 목적으로 고안되었다. 주어진 옵션이 별표(*)가 붙은 `\image*`에 적용된다.

```
\ImageSetup{
  frame=false, scale=1, landscape=false, ...
  star={frame=true, scale=0.5, ...},
  vbar={landscape=true, ...}
}
\image{foo} \image*{foo} \image|{foo} \image*|{foo}
```

`vbar` = 옵션
`star` 옵션과 마찬가지로, 주어진 옵션이 수직선(|)이 붙은 `\image|`에 적용된다.

`\listing` 명령은 목록 환경에서 이미지를 삽입하기 위해 고안되었다. `\image` 명령을 위한 전역 설정들 가운데 `scale`와 `showfilename` 옵션만이 이 명령에 적용된다.

```

\begin{enumerate}
\item 이 명령의 주요 목적은 소프트웨어 설명서에서 어떤 조작에 따르는 결과를 보여주
↪ 는 것이다.
\listing{uncertains}
\item 윗 글줄과 이미지 사이의 간격을 옵션으로 지정할 수 있다.
\listing[-1ex]{uncertains}
\end{enumerate}

```

1. 이 명령의 주요 목적은 소프트웨어 설명서에서 어떤 조작에 따르는 결과를 보여주는 것이다.



2. 윗 글줄과 이미지 사이의 간격을 옵션으로 지정할 수 있다.



\listing*에 대해서는 32 페이지 [가짜 이미지](#)를 보라.

번역사를 위한 이미지 대조

소프트웨어 설명서를 번역할 때 가장 골치아픈 문제는 사용자 인터페이스 스크린샷들이다.

보안 형태 옵션을 개인용으로 설정한 경우에, 사용되는 암호화 방식을 선택하십시오.
If Security Type is set to Personal, select the security type in use.

이 예에서, 번역사에게 넘기기 전에 “보안 형태”를 “Security Type”으로 바꾸어두는 것이 기술적으로 가능하고 가장 바람직하지만, 대개 제공된 문자열 대조표에서 번역사가 찾아 바꾸어야 한다. 엑셀로 작성된 문자열 대조표를 일일이 찾아보는 것은 번잡스럽고 헛갈린다.

별로 쓰일 것 같지 않지만, 번역사들에게 도움이 될까하여 실험적인 \mateimage 명령이 고안되었다.

```

\mateimage[옵션]{이미지 파일}
\MateSetup{옵션}

```

\mateimage의 옵션은 \image 명령에 전달된다. \MateSetup에 지정할 수 있는 옵션들은 다음과 같다.

mate = first, second, both

firstpath = ../images/kor/

한국어 스크린샷 이미지들이 포함된 폴더

```
secondpath = ../images/eng/
```

영어 스크린샷 이미지들이 포함된 폴더

mate에 both가 설정되어 있을 때 `\mateimage{foo.png}`는 `../images/kor/foo.png`와 `../images/eng/foo.png`를 삽입한다. 파일 이름들이 서로 동일해야 한다.

3.2 설명 딸린 이미지

`\illustimage` 명령 또는 `IllustImage` 환경이 이미지와 설명문을 나란히 배치한다.

```
\begin{IllustImage}{uncertainm}
```

이 환경은 주어진 이미지에 대한 짧은 설명문을 곁에 두기 위한 목적으로 고안되었다.

`\macro{wrapfigure}` 환경과 달리, 글이 이미지 높이를 넘쳐도 이미지 아래로 흐르지 않는다.
↔ 는다.

```
\end{IllustImage}
```



이 환경은 주어진 이미지에 대한 짧은 설명문을 곁에 두기 위한 목적으로 고안되었다. `wrapfigure` 환경과 달리, 글이 이미지 높이를 넘쳐도 이미지 아래로 흐르지 않는다.

```
\illustimage*[옵션]{이미지 파일}(캡션){텍스트}
```

```
\begin{IllustImage}*^*[옵션]{이미지 파일}(캡션)
```

설명문

```
\end{IllustImage}
```

```
\IllustImageSetup{옵션}
```

```
beforekip = 0.25\onelineskip
```

그림 위 간격

```
extraskip = 0.5\onelineskip
```

그림 위 간격. 이 값이 샷갓표 (^)가 붙은 `\illustimage^` 또는 `\begin{IllustImage}^`에 적용된다.

```
afterskip = 0.25\onelineskip
```

그림 아래 간격

```
frame = true/false
```

이미지 테두리에 선이 그어진다.

```
gap = 1em
```

이미지와 설명문 사이 간격

`scale` = 1.0
 주어진 배율만큼 이미지가 확대되거나 축소된다.

`position` = left/right
`right`를 지정하면 설명이 왼쪽에, 이미지가 오른쪽에 배치된다.

`voffset` = 0pt
 이미지 상단과 설명문 상단을 정렬하기 위해 설명문 위에 추가되는 간격

`valign` = t/c/b
 이미지와 설명문의 정렬 기준 (상단, 가운데, 하단)

`caption` = 텍스트
 캡션. 파일 이름 따위를 나타내기 위한 목적으로 밑줄 문자(_)를 사용할 수 있다.

`caption-verb` = true/false
 기호 문자들을 백슬래시 없이 그대로 *verbatim* 캡션에 사용할 수 있다.

`caption-ignore` = true/false
 캡션이 식자되지 않는다.

`caption-align` = \raggedright, \centering, \raggedleft
 캡션 정렬

`label` = 텍스트
 상호참조를 위한 라벨. 달리 지정하지 않으면 이미지 파일 이름이 라벨이 된다.

`legend` = true/false
 캡션에 번호가 붙지 않는다.

`minwidth` = 0pt
 이미지 영역의 최소 폭. 0 포인트보다 큰 값이 지정되었을 때 유효하다.

`imagealign` = \raggedright, \centering, \raggedleft
 이미지 영역 안에서 이미지 정렬

`broad` = true/false
 변이단 조판에서 이미지 영역이 왼쪽 여백으로 넘어간다.

`showfilename` = true/false
 이미지 파일의 이름이 표시된다.

`textstyle` = \raggedright\small
 설명문 스타일

`fake` = true/false
 32 페이지 **가짜 이미지**를 보라.

`star` = 옵션
 이것은 기본 설정과 다른 작동을 허용하기 위한 목적으로 고안되었다. 주어

진 옵션이 별표가 붙은 `\illustimage*` 또는 `\begin{IllustImage}*` 에 적용된다.

enum = 옵션
주어진 옵션이 `\begin{IllustEnum}` 에 적용된다.

`IllustEnum`은 설치 절차와 같은 순서를 나타내기 위한 목적으로 `IllustImage`로부터 만들어진 환경이다.

```
\IllustImageSetup{enum=frame}
\begin{IllustEnum}*{uncertainm}
  \item 첫 단계를 나타내기 위해 별표를 붙인다.
\end{IllustEnum}
\begin{IllustEnum}^{uncertainm}
  \item 삿갓표를 붙이면 \macro{star}에 부여된 옵션들이 적용된다.
\end{IllustEnum}
```



1. 첫 단계를 나타내기 위해 별표를 붙인다.



2. 삿갓표를 붙이면 star에 부여된 옵션들이 적용된다.

필요하다면 `IllustImage`를 이용하여 다른 옵션들로 작동하는 환경들을 만들 수 있다.

```
\tl_new:N \l_illust_item_options
\NewDocumentCommand \IllustItemSetup { m }
{
  \tl_set:Nn \l_illust_item_options { #1 }
}
\NewDocumentEnvironment { IllustItem } { 0 } { m +b }
{
  \group_begin:
    \exp_args:No \IllustImageSetup{ \l_illust_item_options }
    \illustimage[#1]{#2}
    {
      \begin{itemize}
```

```

#3
\end{itemize}
}
\group_end:
}{}


```

3.3 글줄 안에 이미지

스마트폰 설명서를 만드는 데에서 성가신 일들 가운데 하나가 버튼이나 아이콘 같은 작은 이미지들을 그려 넣는 것이다. `\img` 명령이 이를 위해 고안되었다.

```
\LineImageSetup{scale=0.75}
```

이 버튼이 `\img[-0.1ex]{button_menu}` 흔히 메뉴를 가리킨다.

| 이 버튼이  흔히 메뉴를 가리킨다.

```

\LineImageSetup{옵션}
\image*[높이]{이미지 파일}

```

`scale` = 1.0
주어진 배율만큼 이미지가 확대되거나 축소된다.

`raise` = -0.25ex
이미지가 기준선 `baseline` 으로부터 주어진 크기만큼 올라가거나 내려간다.

`showfilename` = true/false
이미지 파일의 이름이 표시된다.

`before` = \space
이미지 앞 간격

`after` = \space
이미지 뒤 간격

`\image*`에 대해서는 다음 절을 보라.

3.4 가짜 이미지

```

\fakeimage[text]
\fakeimg[text]

```

이미지 파일이 아직 준비되지 않았을 때 이미지가 들어갈 자리를 나타내기 위해 `\fakeimage`와 `\fakeimg`가 고안되었다. `\image` 명령과 `\listing*` 명령 그리고 `IllustImage` 환경이 `\fakeimage`를 사용하고, `\img*` 명령이 `\fakeimg`를 사용한다.

```

\image[fake]{home_screenshot.png}
\img*{button_share.png}

```


home_screenshot.png

?button_share.png?

3.5 캡션

`\CaptionSetup` 명령을 이용하여 캡션 스타일을 바꿀 수 있다.

```
\CaptionSetup{옵션}
```

`before skip = 1ex`

캡션 위 간격

`after skip = 1ex`

캡션 아래 간격

`align = \raggedright, \centering`

캡션이 두 줄 이상일 때 정렬 방식

`align-short = \centering`

캡션이 한 줄 이하일 때 정렬 방식

`font = \beseries\small`

폰트

`delimiter = :`

번호와 캡션 사이 구획 문자

3.6 콜아웃

콜아웃 `callout` 은 명칭이나 기능을 설명하기 위해 삽화에 덧붙인 글이나 번호를 말한다. 텍스트 콜아웃이 번호 콜아웃보다 독자에게 더 친절한 것이지만, 삽화에 덧붙여진 텍스트를 수정하거나 번역하는 일은 불편하다. 그래서 대부분의 테크니컬 라이터들이 번호 콜아웃을 선호한다.

```
\begin{callout}*[칼럼 수] \item \end{callout}
\CalloutSetup{옵션}
```

`callout` 환경이 번호 콜아웃 목록을 만든다.

```
\begin{callout}
\item Earpiece
```

```

\item Proximity sensor
\item Front camera
\item Volume button
\item Touchscreen
\item Microphone
\end{callout}

```

- | | | |
|------------------|---------------------|-----------------|
| 1) Earpiece | 2) Proximity sensor | 3) Front camera |
| 4) Volume button | 5) Touchscreen | 6) Microphone |

\CalloutSetup 명령을 이용하여 콜아웃 스타일을 바꿀 수 있다.

beforeskip = \hnullldim, 0.5\onelineskip

목록 위 간격. 값이 \hnullldim이면 \ObjectSetup으로 설정된 값이 사용된다. 55 페이지 [개체](#)를 보라.

afterskip = \hnullldim, 0.5\onelineskip

목록 아래 간격.

rule = true/false

목록 위와 아래에 선이 그어지고, 결과적으로 목록이 표처럼 보인다.

column = 3

칼럼 수

font = \small

폰트

label = \makebox[1.25em]{\hfill\calloutno}}\enspace

번호 스타일.

tab = true/false

이 옵션이 false로 설정되면 칼럼 간격이 무시된다.

space = 0.25em

tab 옵션이 false일 때 이 값이 항목 간격이 된다.

다음과 같은 방법으로 각 칼럼의 시작 위치를 조정할 수 있다.

```

\begin{callout}\TabPositions{0pt, .425\linewidth, ...}

```

별표가 붙은 \begin{callout}*은 enumerate의 변형인 calloutenum 환경을 부른다. 콜아웃 번호를 원 숫자로 표현하려면 다음과 같이 설정한다.

```

\CalloutSetup{label=\cirnum{\calloutno}\enspace}
\setlist[calloutenum]{label=\cirnum{\arabic*}, noitemsep}

```

\cirnum에 대해서는 48 페이지 [원 숫자](#)를 보라.

3.7 이미지 파일 이름과 번호

텍스트로 만들어진 문서를 인디자인 같은 다른 프로그램의 형식으로 옮겨야 할 때, 이미지 아래에 표시된 파일 이름이 이미지들을 식별하는 데에 도움이 된다.

```
\ShowImageFilename*
\ShowImageCounter*
\ResetImageCounter
```

\ShowImageFilename 명령은 \ImageSetup과 \IllustImageSetup에, \ShowImageFilename*은 \LineImageSetup까지 포함하여, showfilename 옵션을 지시한다.

\ShowImageCount 명령은, 파일 이름 대신, 이미지 아래에 일련번호를 붙인다. 이 명령은 문서에 포함된 많은 이미지들의 수를 파악하기 위해 고안되었다. \ResetImageCounter 명령은 일련번호를 “0”으로 되돌린다.

3.8 앨범 만들기

```
\MakeAlbum*[이미지 배열]{foo.txt}
```

```
foo.txt:
image_01.jpg
image_02.png
image_03.pdf
```

\MakeAlbum은, 위 예와 같이, 텍스트 파일로부터 이미지 목록을 읽고 차례로 이미지들을 삽입한다. \MakeAlbum*은 이미지 파일의 이름을 식자하지 않는다. 다음 예는 현재 폴더에서 이미지 파일들을 모아 앨범을 만드는 간단한 파워셸 PowerShell 스크립트이다.

```
$dir = "images.txt"
$tex = "album.tex"
$imageTypes = @("pdf", "jpg", "png")
foreach ($element in $imageTypes) {
    Get-ChildItem ".*$element" -name | Add-Content $dir -Encoding UTF8
}
$src = "\documentclass{hzguide}
\LayoutSetup{paper=A4}
\begin{document}
\MakeAlbum{$dir}
\end{document}"
Set-Content $tex -Encoding UTF8 $src
xelatex.exe -interaction=batchmode $tex
```

3.9 표

Table 환경이 표의 스타일을 제어한다.

```

\TableSetup{옵션}
\begin{Table}*[옵션](표 제목)<각주>
\begin{tblr}{lXX}
...
\end{tblr}
\end{Table}

```

일반 출판물과 달리 설명서에서 표는 보충적인 자료가 아니라 핵심적인 정보의 일부이기 때문에 절 제목 아래에 오는 표 하나가, 다른 텍스트 없이도, 그 절의 온전한 내용이 될 수 있다. 따라서 불가피한 경우가 아니라면 표에 제목을 붙이지 않는 것이 바람직하다.

각주를 다는 방법에 대해 37 페이지 [제품 사양을 보여주는 표](#)를 보라.

Table 환경과 TableSetup 명령에 다음과 같은 옵션들을 사용할 수 있다. tabulararray 패키지를 사용하는 경우에 많은 옵션들이 무용하지만, tabular나 tabularx 환경에는 유용할 것이다.

beforekip = \hznulldim, 0.5\onelineskip

표 위 간격. 값이 \hznulldim이면 \ObjectSetup으로 설정된 값이 사용된다.
55 페이지 [개체](#)를 보라.

afterskip = \hznulldim, 0.5\onelineskip

표 아래 간격.

float = true/false

이 옵션은 표 내용을 table 환경에 넣는다. 다시 말해 표의 위치가 텍에 의해 결정된다.

float-placement = tbh

table 환경을 이용하는 경우에 위치 옵션

align = \raggedleft, \centering, \raggedright

표 정렬

landscape = true/false

표가 90도 회전한다. 캡션도 함께 회전한다.

caption = 텍스트

캡션

label = 텍스트

상호참조를 위한 라벨

legend = true/false

캡션에 번호가 붙지 않는다.

font = \rmfamily

폰트

fontsize = \small

폰트 크기

footnotestyle = arabic, fnsymbol

각주 번호 스타일

stretch = 1.0

줄 간격. 이것은 \arraystretch를 설정하는 것과 같다.

star = 옵션

이것은 기본 설정과 다른 작동을 허용하기 위한 목적으로 고안되었다. 주어진 옵션이 별표가 붙은 \begin{Table}*에 적용된다.

```
\TableSetup{
  font=\rmfamily,
  fontsize=\small, ...,
  star={font=\sffamily, ...},
  vbar={font=\footnotesize, ...}
}
\begin{Table}
\begin{Table}*
\begin{Table}|
\begin{Table}*|
```

vbar = 옵션

star 옵션과 마찬가지로, 주어진 옵션이 수직선이 붙은 \begin{Table}|에 적용된다.

제품 사양을 보여주는 표

SpecTable 환경은 설명서에서 빈번하게 사용되는 표 형식 중 하나인 제품 사양을 보여주기 위한 장치이다. \TableSetup에 의한 전역 설정이 이 환경에 유효하다.

```
\begin{SpecTable}*[옵션](캡션)<각주>
...
\end{SpecTable}
```

```
\begin{SpecTable}<
\footnotetext[1]{If the temperature inside the battery pack rises ...}
\footnotetext[2]{As the degradation of batteries is accelerated ...}
>
Nominal voltage\footnotemark[1] & 51.8 V \\
Operating voltage & 45.2 V to 58.1 V \\
Nominal capacity\footnotemark[2] & 126 A·h \\
Nominal energy & 6.4 kW·h
\end{SpecTable}
```

Nominal voltage ¹	51.8 V
Operating voltage	45.2 V to 58.1 V
Nominal capacity ²	126 A·h
Nominal energy	6.4 kW·h

¹If the temperature inside the battery pack rises ...

²As the degradation of batteries is accelerated ...

별표 옵션(`\begin{SpecTable}`*)을 사용하면 첫 줄이 제목 줄로 간주된다.

제품 구성품을 보여주는 표

`ImageTable` 환경은 제품과 함께 제공되는 액세서리를 보여주기 위한 장치이다. `\TableSetup`에 의한 전역 설정이 이 환경에도 적용된다.

```
\begin{ImageTable}*<각주>
```

```
...
```

```
\end{ImageTable}
```

```
\begin{ImageTable}
```

```
\img{uncertains} & \img{uncertains} & \img{uncertains} \\\
```

```
액세서리 & 액세서리 & 액세서리 \\\
```

```
\end{ImageTable}
```



액세서리



액세서리



액세서리

별표 옵션(`\begin{ImageTable}`*)을 사용하면 칼럼 수가 둘로 줄어든다.

3.10 표 안에 이미지

```
\CellImageOffset{1.5\onelineskip}
```

```
\cellimg[높이]{이미지 파일}
```

표 안에 이미지를 넣으면 다른 셀들의 기준선이 이미지에 맞춰지는 문제가 발생한다. 이를 피하기 위해 `\cellimg` 명령이 고안되었다. 주어진 높이만큼 이미지가 올라가거나 내려간다. 디폴트 높이를 `\CellImageOffset` 명령으로 바꿀 수 있다.

아래 예가 `\cellimg` 명령을 이용하는 테이블 환경이다.

```

\NewDocumentEnvironment{SignTable}{+b}
{
  \begin{longtblr}{
    colspec=lX,
    hline{1-Z}=0.3pt,
    column{2}={font=\rmfamily\normalsize},
    rowsep=2pt,
    stretch=1.5
  }
  #1
  \end{longtblr}
}{}

\begin{SignTable}
\celling{foo} & ...
\end{SignTable}

```


제 4 장

목록과 용어

4.1 나열과 절차	41
4.2 정의 목록	42
4.3 선택 목록	44
4.4 용어	45
4.5 첨자	46
4.6 색인	47
4.7 원 숫자	48
4.8 기호	49

4.1 나열과 절차

`itemize`와 `enumerate` 환경을 사용할 때 `memoir` 클래스가 제공하는 `\tightlist` 명령이나 `enumitem` 패키지가 제공하는 `noitemsep` 옵션을 사용하여 항목 간격을 줄일 수 있다. 편의를 높이고자 별표를 사용하여 이를 지시할 수 있도록 `itemize` 및 `enumerate` 환경이 재정의되었다.

```
\begin{enumerate}\tightlist
\begin{itemize}[noitemsep]
\begin{enumerate}*
\begin{itemize}*

```

필요한 경우가 드물겠지만, `\SetListStyle` 명령을 이용하여 이 두 환경에 서로 다른 스타일을 적용할 수 있다.

```
\SetListStyle{
  enumerate={\raggedright\small}
  itemize={\sffamily}
}

```

4.2 정의 목록

애플리케이션 프로그램의 설정 옵션들과 같은 것들을 설명하는 데에 `description` 환경이 사용된다. `enumitem` 패키지가 제공하는 `\newlist` 명령을 이용하여 다른 형태의 정의 목록을 만들 수 있지만 흡족하지 않다.

`\NewTerms` 명령은 다양한 형태의 정의 목록을 만드는 장치이다.

```
\NewTerms{환경 이름}{옵션}  
\RenewTerms{환경 이름}{옵션}  
\TermsSetup[환경 이름]{옵션}
```

`\TermsSetup`에 환경 이름을 지정하지 않으면 `\NewTerms`에 의해 만들어진 모든 환경에 주어진 설정이 적용된다.

`labeltype` = default, singleline, multiline, macro, macrosingle, adhoc

`singleline`을 지정하면 표제어가 한 줄을 차지한다. `macro`는 백슬래시를 비롯한 기호 문자들을 허용한다. 이 두 가지를 결합한 것이 `macrosingle`이다. `multiline`을 지정하면, 표제어가 주어진 폭보다 길 때 두 줄이 된다. 65 페이지 [소스 코드](#)를 보라.

`adhoc` 라벨 유형은 정의되어 있지 않다. 다음과 같이 그것을 활용하여 새 유형을 만들 수 있다.

```
\cs_new:Npn \terms_label_adhoc:n #1  
{  
  \macro[#1]{<#1>  
  \l_terms_delimiter_tl  
}  
\NewTerms{Tags}{labeltype=adhoc, delimiter=\hfill}
```

`offset` = 0em

다른 목록 환경 아래에서 왼쪽 여백에 추가되는 길이

`marker` = { }, \textbullet

표제어 앞에 붙이는 기호 문자. 앞의 것이 상위 수준에, 뒤의 것이 하위 수준에 적용된다.

`markersep` = 0.5em

기호 문자 뒤의 간격

`enumerate` = true/false

기호 문자 대신 번호가 붙는다.

`enummarker` = {\int_use:N \l_terms_int)\hskip\l_terms_marker_sep_skip}

번호 형식

`base` = MM

주어진 문자의 길이 만큼 표제어 영역의 폭이 정해진다.

`font` = \bfseries

표제어 폰트

<code>color</code>	<code>= black</code> 표제어 색
<code>delimiter</code>	<code>= :\hfill</code> 표제어 뒤에 구획 문자
<code>labelwidth</code>	<code>= 2em</code> 표제어 영역의 폭
<code>labelvoffset</code>	<code>= 0pt</code> 표제어 위에 추가되는 간격. 이것은 <code>labeltype</code> 옵션에 <code>multiline</code> 이 설정된 경우에만 유효하다.
<code>labelsep</code>	<code>= 0.5em</code> 표제어 뒤의 간격
<code>itemindent</code>	<code>= 0em</code> 표제어 뒤에 들여쓰기 폭
<code>leftmargin</code>	<code>= 0pt</code> 왼쪽 여백
<code>index</code>	<code>= true/false</code> 표제어가 색인에 추가된다.
<code>indexcategory</code>	<code>= 단어</code> 표제어가 색인에서 주어진 단어 아래에 들어간다.
<code>indexsame</code>	<code>= true/false</code> 표제어가 지정된 것과 동일한 폰트로 색인에서 식자된다.
<code>topsep</code>	<code>= 1.25\onelineskip, 3ex</code> 목록 위 간격
<code>itemsep</code>	<code>= 0.25\onelineskip, 1ex</code> 항목들 사이 간격
<code>style</code>	<code>= \raggedright, ...</code> 설명문 스타일
<code>beforelist</code>	<code>= \RuleBox, \FrameBox, ...</code> 환경 앞에 추가되는 명령. 이를테면 이 옵션과 아래 옵션에 <code>FrameBox</code> 환경의 시작 명령과 종료 명령을 지정함으로써 목록을 상자 안에 넣을 수 있다.
<code>afterlist</code>	<code>= \endRuleBox, \endFrameBox, ...</code> 환경 뒤에 추가되는 명령
<code>star</code>	<code>= 옵션</code> 이것은 기본 설정과 다른 작동을 허용하기 위한 목적으로 고안되었다. 주어진 옵션이 별표가 붙은 <code>\begin{LIST}*</code> 에 적용된다.
<code>vbar</code>	<code>= 옵션</code> <code>star</code> 옵션과 마찬가지로, 주어진 옵션이 수직선이 붙은 <code>\begin{LIST} </code> 에 적용된다.

`\NewTerms` 명령으로 다음과 같은 환경들이 저마다의 목적으로 정의되어 있다.

```
\begin{terms}*[옵션](표제어 폭)<상위 색인> \item[] \end{terms}
\begin{UI}*[옵션](표제어 폭)<상위 색인> \item[] \end{UI}
\begin{signs}*[옵션](표제어 폭)<상위 색인> \item[] \end{signs}
```

(MMM)을 지정하는 것은 `base=MMM` 또는 `labelwidth=4em`을 설정하는 것과 마찬가지이다. <단어>를 지정하는 것은 `indexcategory=단어`를 설정하는 것과 같다.

terms	전문 용어 <code>jargon</code>
UI	사용자 인터페이스 문자열
signs	안전 표지 같은 이미지. 이 환경에서 <code>\item[]</code> 대신 <code>\imgitem[]</code> 을 사용해야 한다.

표제어들 가운데 가장 긴 것으로 `base` 옵션을 설정하는 것은 성가신 일이다. 그 수고를 덜어주고자 `\CloneTerms` 명령이 고안되었다.

```
\CloneTerms{새 이름}{기존 이름}

\CloneTerms{Foo}{foo}
\begin{Foo}[옵션] \item[] \end{Foo}
```

`\CloneTerms` 명령으로 만들어진 새로운 환경은 주어진 모든 표제어의 길이를 재고 가장 긴 값을 구하여 기존 환경에 넘긴다. 새 환경의 형식은 원래의 것보다 단순하여 별표와 수직선 인자를 사용할 수 없다.

```
\CloneTerms{Terms}{terms}
\begin{Terms}
\item[Volumes] Adjust the volume for ...
\item[Silent mode] Set to vibrate or mute ...
\item[Default notification] Select a sound for ...
\end{Terms}
```

Volumes:	Adjust the volume for ...
Silent mode:	Set to vibrate or mute ...
Default notification:	Select a sound for ...

4.3 선택 목록

`options` 환경을 이용하여 지원되는 기능들이나 호환되는 제품들과 같은 목록을 만들 수 있다.

```
\begin{options}*[항목 번호, ...]
\item ...
\end{options}
```

별표가 붙은 `\begin{options}*`는 항목 사이 간격을 넓힌다.

This edition supports as follows:

```
\begin{options}[2,4]
\item Syntax highlighting
\item User snippets
\item Keyboard shortcuts
\item Color themes
\end{options}
```

This edition supports as follows:

- ☐ Syntax highlighting
- ☒ User snippets
- ☐ Keyboard shortcuts
- ☒ Color themes

라벨들을 다른 기호로 바꾸려면 `\optselected`와 `\optunselected`를 재정의하라. 라벨들은 `\textsb` 명령으로 식자된다. 49 페이지 [기호](#)를 보라.

4.4 용어

`\NewTerm` 명령을 이용하여 버튼 라벨 같은 말들을 강조하거나 색인에 넣는 명령들을 만들 수 있다.

```
\NewTerm{명령 이름}[옵션]
\RenewTerm{명령 이름}[옵션]
\TermSetup[명령 이름][옵션]
```

`\TermSetup`에 명령 이름을 지정하지 않으면 `\NewTerm`에 의해 만들어진 모든 명령에 주어진 설정이 적용된다.

`font` = `\sffamily\bfseries`
용어 폰트

`color` = `black`
용어 색

`index` = `true/false`
용어가 색인에 추가된다.

`indexsame` = `true/false`
표제어가 색인에서 지정된 것과 동일한 폰트로 식자된다.

`breakable` = `true/false`
`true`이면 줄 나눔이 허용된다.

`star` = 옵션
이것은 기본 설정과 다른 작동을 허용하기 위한 목적으로 고안되었다. 주어진 옵션이 별표가 붙은 `\TERM*`에 적용된다.

vbar = 옵션

star 옵션과 마찬가지로, 주어진 옵션이 수직선이 붙은 \TERM! 에 적용된다.

\NewTerm 명령으로 다음과 같은 명령들이 저마다의 목적으로 정의되어 있다.

```
\term*!{색인 정렬}[단어][상위 색인]
\ui*!{색인 정렬}[단어][상위 색인]
\mi*!{색인 정렬}[단어][상위 색인]
```

\term[csharp]{C\#}[language] 는 \index{csharp@C\#} 그리고 \index{language!csharp@C\#}을 만들어낸다.

\term 전문 용어 jargon

\ui 사용자 인터페이스 문자열

\mi 기기의 지시등이나 버튼에 표시된 문자열


\Menu 명령은 menukeys 패키지의 \menu 명령이 하는 것과 동일하다. 다른 점은 마지막 항목을 색인에 추가하는 것이다.

```
\Menu{File>Preferences>{User Snippets}>{latex.json}}
```

File > Preferences > User Snippets > latex.json

tabto 패키지의 \tab 명령과의 충돌을 회피하기 위해 \Tab 키가 정의되어 있다.

스니펫으로 바꾸려면 \keys{\Tab} 키를 눌러라.

스니펫으로 바꾸려면  키를 눌러라.

url 패키지의 \path 명령을 써서 파일 경로를 나타낼 수 있다. 그런데 이 명령이 \footnote 같은 명령에 포함될 때 언더스코어 같은 기호 문자 앞에 공백이 생긴다. 대안으로 \winpath가 만들어졌다.

settings.json 파일이 \winpath{C:\Users\USERNAME\AppData\Roaming\Code\User} 아래에
→ 저장된다.

settings.json 파일이 C:\Users\USERNAME\AppData\Roaming\Code\User 아래에 저장된다.

4.5 첨자

\textsuperscript와 \textsubscript의 다른 이름으로서 \Sup와 \Sub가 정의되어 있다. 화학식과 단위들은 정자로 표기되어야 하는데 수식을 이용하여 그것들을 표기하는 것이 도리어 번잡하기 때문이다.

```
$\mathrm{NH}_3$    NH\Sub{3}
$\mathrm{\mu g/m}^3$    \mu g/m\Sup{3}
```

NH₃ NH₃ μg/m³ μg/m³

`\annotate` 명령은 `\textsuperscript` 및 `\textsubscript`를 대체하기 위해 고안되었다.

```
\AnnoateSetup{옵션}
\annotate*[옵션]{주석 텍스트}
\anota*[옵션]{주석 텍스트}
```

`\anota`는 `\annotate`의 다른 이름이다.

`breakable` = true/false

이 옵션이 false이면 `\textsuperscript`와 `\textsubscript`가 사용된다. 이것은 줄 나눔이 허용되지 않는다는 것을 의미한다.

`superscript` = true/false

true이면 위 첨자로, false이면 아래 첨자로 식자된다.

`opening` = (

주석 앞에 오는 문장 부호나 기호 문자

`closing` =)

주석 뒤에 오는 문장 부호나 기호 문자

`font` = `\scriptsize`

주석 폰트

`color` = darkgray

주석 색

`space` = `\thinspace`

주석 앞과 뒤에 오는 공백 문자

`raise` = 0.75ex

주석이 기준선으로부터 주어진 크기만큼 올라간다. `superscript` 옵션이 false이면 내려간다.

`index` = true/false

주석이 색인에 추가된다.

`star` = 옵션

이것은 기본 설정과 다른 작동을 허용하기 위한 목적으로 고안되었다. 주어진 옵션이 별표가 붙은 `\annotate*`에 적용된다.

4.6 색인

HzGuide 클래스는, 다양한 언어들의 조판을 목적으로 하기 때문에, 색인 정렬에 `xindex`나 `texindy`를 사용할 것을 권장한다.

```
\IndexingEnable*
\IndexingDisable
\BookmarkIndexHead
```

`\IndexingEnable`은 앞에서 설명한 색인을 지원하는 명령과 환경들의 `index` 옵션을 true로 변경한다.

- \NewTerm 명령에 의해 만들어진 명령들
- \NewTerms 명령에 의해 만들어진 환경들
- \annotate 명령
- \macro 명령. 65 페이지 [소스 코드](#)를 보라.

\BookmarkIndexHead 명령은 색인 페이지에서 한글 자모 또는 알파벳을 가리키는 북마크를 PDF에 추가한다. 별표가 붙은 \IndexingEnable* 명령이 \BookmarkIndexHead 명령을 호출한다.

4.7 원 숫자

콜아웃 번호를 표현하기 위해 고안된 \cirnum 명령을 이용하여 원 숫자 또는 원 문자를 만들 수 있다.

```
\CirnumSetup{옵션}
\cirnum{숫자}
```

```
\CirnumSetup{
  font=\small\sffamily\bfseries,
  fgcolor=white,
  bgcolor=MidnightBlue,
  sep=1pt,
  raise=-.5ex,
  base=99
}
\cirnum{1} \cirnum{20} \cirnum{A} \cirnum{a} \cirnum{가} \cirnum{ㄱ}
```

1 20 A a 가 ㄱ

fgcolor = white
글자 색

bgcolor = black
바탕 색

font = \small\sffamily\bfseries
글자 색을 하양으로 지정한다면 이 옵션의 설정에 \bfseries를 포함시키는 것이 좋다.

fontfeature = \addfontfeature{LetterSpace=-3}
폰트에 추가할 특성

base = 99
주어진 글자의 길이 만큼 동그라미의 폭이 정해진다.

sep = 1pt
동그라미와 글자 사이 간격

`raise = -0.5ex`

글자가 기준선으로부터 주어진 크기만큼 올라가거나 내려간다.

4.8 기호

다양한 기호들이 종종 설명서에 필요한 반면, 지름 기호나 체크 박스까지 포함하는 폰트들은 흔하지 않다.

```
55 \newfontfamily\sbfamily{HANDotum.ttf}[BoldFont=HANDotumB.ttf]
56 \DeclareTextFontCommand{\textsb}{\sbfamily}
```

오로지 기호들을 식자하기 위한 목적으로 `\sbfamily`와 `\textsb` 명령이 정의되어 있다. 디폴트 폰트는 함초롬 돋움이다.

```
\textsb{ } \textsb{☑}
```

| ∅ ☑

제 5 장

글상자와 경고문

5.1 글상자	51
5.2 경고문 표어	54

5.1 글상자

tcolorbox 패키지에 기반하여 여러 모양의 박스들이 정의되어 있다.

```
\begin{FrameBox} \end{FrameBox}
```

```
\begin{LeftBarBox} \end{LeftBarBox}
```

```
\begin{RightBarBox} \end{RightBarBox}
```

제목

```
\begin{ShadeBox}[제목] \end{ShadeBox}
```

제목

```
\begin{ShadowBox}[제목] \end{ShadowBox}
```

```
\begin{RuleBox} \end{RuleBox}
```

제목

```
\begin{TabBox}[\ 제목\ ] \end{TabBox}
```

제목

```
\begin{TabRuleBox}[\ 제목\ ] \end{TabRuleBox}
```

이 박스들은 직접 사용되기보다, 무엇보다도 위와 아래 간격들이 0 포인트로 설정되어 있기 때문에, `\NewBoundedBox` 명령이 정의하는 환경에서 사용될 목적으로 만들어졌다.

```

\NewBoundedBox{환경 이름}[옵션]
\RenewBoundedBox{환경 이름}[옵션]
\BoundedBoxSetup[환경 이름][옵션]

```

\BoundedBoxSetup에 환경 이름을 지정하지 않으면 \NewBoundedBox에 의해 만들어진 모든 환경에 주어진 설정이 적용된다.

beforeskip = \hznulldim, 0.25\onelineskip

박스 위 간격. 값이 \hznulldim이면 \ObjectSetup으로 설정된 값이 사용된다. 55 페이지 [개체](#)를 보라.

extraskip = -0.5\onelineskip

박스 위 간격. 이 값이 샷갓표(^)가 붙은 \begin{BOX}^에 적용된다.

afterskip = \hznulldim, 0.25\onelineskip

박스 아래 간격.

type = FrameBox, LeftBarBox, RightBarBox, ShadeBox, ShadowBox, RuleBox, TabBox, TabRuleBox

박스 모양. 디폴트는 ShadeBox이다.

symbol = \img{foo}\space

제목 앞에 오는 이미지

signal = 표어

특정한 목적을 위해 문서 전반에 걸쳐 사용되는 “경고”나 “주의” 같은 경고 문 표어

beforesignal = <, \space

제목 앞에 오는 문자

aftersignal = >, \space

제목 뒤에 오는 문자

style = \raggedright

문단 스타일

titlefont = \bfseries

제목 폰트

breakable-default = true/false

이 옵션이 true이면 박스의 쪽 나눔이 허용된다.

breakable-adhoc = true/false

이 옵션은 breakable-default에 설정된 것과 반대로 설정되어야 한다. 이 설정이 별표가 붙은 \begin{BOX}*에 적용된다.

아래 예가 \NewBoundedBox 명령으로 만들어진 환경의 형식을 보여준다.

```

\NewBoundexBox{advice}[옵션]

\begin{advice}*^[옵션](제목)
...

```

```
\end{advice}
```

\NewBoundedBox 명령으로 다음과 같은 환경들이 저마다의 목적으로 정의되어 있다. 심각성에 따른 경고문 admonition 유형들은 IEC/ISO 82079나 ASD-STE100 같은 표준에 따라 약간 다르다.

위험

위험은 사망이나 심각한 부상을 초래하는 상황을 가리킨다. danger 환경이 이렇게 정의되어 있다.

```
4324 \NewBoundedBox{danger}{signal=\g_admonition_danger_tl,  
↪ symbol={\img{AlertSymbol}~}}
```

경고

경고는 사망이나 심각한 부상을 초래할 수 있는 상황을 가리킨다. warning 환경이 이렇게 정의되어 있다.

```
4325 \NewBoundedBox{warning}{signal=\g_admonition_warning_tl,  
↪ symbol={\img{AlertSymbol}~}}
```

주의

주의 caution 는 경미한 부상을 초래하는 상황을 가리킨다. caution 환경이 이렇게 정의되어 있다.

```
4326 \NewBoundedBox{caution}{signal=\g_admonition_caution_tl}
```

주의

주의 notice 는 물적 피해를 초래하는 상황을 가리킨다. notice 환경이 이렇게 정의되어 있다.

```
4327 \NewBoundedBox{notice}{signal=\g_admonition_notice_tl}
```

참고

참고는 주로 절차에서 추가 정보를 제공한다. note 환경이 이렇게 정의되어 있다.

```
4328 \NewBoundedBox{note}{signal=\g_admonition_note_tl}
```

reference 환경은 파일 경로 같은 것을 예시하기 위한 것으로 이렇게 정의되어 있다.

```
4329 \NewBoundedBox{reference}{style=\small}
```

\RenewTColorBox 명령을 이용하여 이 박스들을 재정의할 수 있다.

```
\RenewTColorBox{ShadeBox}{ s 0{ } }  
{
```

```

...
title=#2,
IfBooleanTF={#1}{...}{...}
}

```

또한 아래 예와 같이, 새로운 박스를 만들고 그 박스를 이용하는 환경을 만들 수 있다.

```

\NewTColorBox{MyBox}{s 0}{ }
{
...
}
\NewBoundedBox{MyBoxEnv}{
type=MyBox,
...
}

```

두 글상자를 중첩하여 사용하는 것은 가능하지만, `\NewBoundedBox` 명령으로 만든 두 환경의 중첩은, 그런 일이 필요할 리 없겠으나, 기대한 결과를 만들어내지 않는다.

5.2 경고문 표어

`\RenewBoundedBox` 명령을 이용하여 경고문 표어 `signal word` 들을 변경하는 것이 번거롭기 때문에 `\AdmonitionSetup` 명령이 고안되었다. 이를테면, `language` 클래스 옵션에 `german` 이 지정되면 다음과 같은 선언이 발효된다.

```

\AdmonitionSetup{
danger=GEFAHR,
warning=WARNUNG,
caution=VORSICHT,
notice=HINWEIS,
note=ANMERKUNG
}

```

제 6 장

문서 제어 도구

6.1 개체	55
6.2 상호 참조와 하이퍼링크	56
6.3 확정되지 않은 사항들	56
6.4 TSV 또는 CSV 파일 읽기	57
6.5 싱글 소스 퍼블리싱	58
조건식	59
이미지 경로	60
텍 파일 경로	62
파일 외부에서 설정 변경하기	63

6.1 개체

\ObjectSetup에 의한 설정이, 일괄적으로 제어할 수 있는 개체들로서, 다음 명령들과 환경들에 영향을 미친다.

- \image 명령
- IllustImage 및 IllustEnum 환경
- callout 환경
- Table 환경
- \NewBoundedBox에 의해 생성된 caution을 비롯한 환경들
- \action 명령

beforeskip = 0.5\onelineskip

개체 위 간격. 예를 들어, \ImageSetup의 beforeskip 옵션에 \hnullldim이 설정되어 있다면, 그 설정이 무시되고 이 옵션에 주어진 값이 사용된다. \hnullldim은 -1000pt로 정의되어 있다.

afterskip = 0.5\onelineskip

개체 아래 간격

framecolor = gray

테두리 색. 이를테면 \image 명령의 frame 옵션에 이 설정이 적용된다.

align = \raggedright
정렬 방식

하나의 명령으로 개체들의 간격을 일관되게 조정한다는 것은 순진한 발상이다. 이를테면 연속된 `IllustEnum` 사이의 간격은 다른 개체들 사이의 간격보다 좁아야 적절하다. 그리고 개체들을 조합하여 만들 수 있는 경우의 수가 많기 때문에 각 개체의 간격을 아무리 세심하게 설계해도 부적절한 간격들이 종종 불가피하게 발생한다. 그런 경우에 조금이나마 수고를 줄이고자 다음 명령들이 고안되었다.

```
\skiplines[-0.75]  
\EnlargePage[1.5]
```

`\skiplines` 및 `\EnlargePage` 명령은 `\onelineskip` 한 줄의 크기를 단위로 행간을 조절한다. `\skiplines[-0.75]`는 `\vspace{-0.75\onelineskip}`과 같고, `\EnlargePage[1.5]`는 `\enlargethispage{1.5\onelineskip}`과 같다.

6.2 상호 참조와 하이퍼링크

페이지 번호와 절 제목을 포함하는 상호 참조들을 서로 다른 색으로 표시하는 것이 합리적이라고 할 수 없다.

```
\DecolorHyperlinks[색][폰트]  
\DecolorHyperlinks*[색]
```

`\DecolorHyperlinks` 명령은 주어진 색과 폰트를 `\titleref` 명령에 의해 표시되는 제목 참조에 적용하고, 페이지 번호를 비롯한 다른 상호 참조에는 아무 색도 할당하지 않는다. 디폴트 색은 `darkgray`이다. 별표가 붙은 `\DecolorHyperlinks*` 명령은 주어진 색을 모든 상호 참조에 적용한다.

아크로벳 리더 Acrobat Reader와 달리 수마트라 PDF Sumatra PDF는 아래 주소를, “https://”를 빠뜨렸기 때문인 듯한데, 제대로 연결하지 못한다.

```
\url{www.daum.net}
```

이 사소하고 성가신 문제를 피하기 위하여 `\URL` 명령과 `\email` 명령이 고안되었다.

```
\URL{웹 주소}  
\email{메일 주소}
```

`\URL`은 “https://”를, 별표가 붙은 `\URL*`은 “http://”를 하이퍼링크에 추가한다. `\email` 명령은 마찬가지로 “mailto”를 하이퍼링크에 추가한다.

6.3 확정되지 않은 사항들

작성 중인 문서에 포함해야 할지 또는 그것이 정확한 사실인지 확신할 수 없을 때 그 사항들을 도드라져 보이게 하고자 `\pending` 명령과 `Pending` 환경이 고안되었다.


```
\pending{텍스트}
\begin{Pending} 텍스트 \end{Pending}
\PendingSetup{옵션}
```

\PendingSetup으로 다음과 같은 옵션들을 변경할 수 있다.

color = purple
글자 색

font = \small
폰트

hide = true/false

이 옵션이 true이면 주어진 텍스트가 식자되지 않는다. \ShowPending과 \HidePending 명령으로 이를 보다 쉽게 제어할 수 있다.

```
\query{텍스트}[짧은 텍스트]
\listofqueries
```

\PendingSetup의 설정이 \query 명령에도 적용된다. 이 명령은 이해당사자들, 특히 제품 개발자들에게 질문할 것들을 표시하기 위한 목적으로 고안되었다. 차이점은 주어진 텍스트 앞에 일련 번호가 붙는다는 것과 차례를 만들 수 있다는 것이다. “짧은 텍스트”가 주어지면 질문 차례에 들어간다. \listofqueries가 질문 차례를 만든다.

6.4 TSV 또는 CSV 파일 읽기

\ReadTSV 명령은 TSV tab-separated values 또는 CSV comma-separated values 파일을 읽고 지정된 스타일로 식자한다.

```
\ReadTSV{파일}[옵션]
\TSVsetup{옵션}
```

파일을 읽을 때 빈 줄들은 무시된다.

```
\begin{filecontents*}{example.csv}
가팀
가나다,abc@xyz.com,010-1234-5678,02-1234-5678
가나라,abd@xyz.com,010-1234-5679,02-1234-5679
나팀
가나마,abe@xyz.com,010-1234-5670,02-1234-5670
가나바,abf@xyz.com,010-1234-5671,02-1234-5671
\end{filecontents*}
\ReadTSV{example.csv}[
  csv=true,
  heading=\subsubsection,
  space=\tab,
  tabs={0pt, 0.2\linewidth, 0.5\linewidth, 0.75\linewidth},
```

```
renderers={\textbf, \textsf}
]
```

가팀

가나다	abc@xyz.com	010-1234-5678	02-1234-5678
가나라	abd@xyz.com	010-1234-5679	02-1234-5679

나팀

가나마	abe@xyz.com	010-1234-5670	02-1234-5670
가나바	abf@xyz.com	010-1234-5671	02-1234-5671

csv = true/false
CSV 파일을 읽으려면 이 옵션을 true로 설정해야 한다.

heading = \subsection
한 칼럼만 가진 줄은 이 옵션에 주어진 명령으로 식자된다.

space = \tab, \space
각 칼럼 사이를 벌리는 간격 명령

tabs = 0pt, 10em, 0.5\linewidth, ...
각 칼럼에 대응하는 위치. space 옵션에 \tab이 설정되지 않으면 이 옵션이 무시된다.

renderers = \textbf, \textsf, ...
각 칼럼에 적용할 명령. 네 칼럼이 있는 경우에 두 가지 명령만 주어진다면 셋째 칼럼과 넷째 칼럼에는 \texttsv 명령이 사용된다. 이 명령은 아무 기능도 하지 않는다. 필요에 따라 재정의하라.

verbatim = true/false
백슬래시를 비롯한 모든 문자들이 그대로 식자된다. 디폴트는 false이다.

interline = \dotfill\linebreak
줄 사이에 넣을 점선 같은 장식을 지정하라.

6.5 싱글 소스 퍼블리싱

거의 모든 기업들이 기능과 성능에서 조금씩 다른 여러 제품들을 구비하고 있다. 하나의 제품으로 모든 구매자의 욕구를 충족시키는 것이 불가능하기 때문이다. 복잡한 제품군을 유지하는 것은 불가피하게 생산 효율을 떨어뜨리고 비용을 증가시킨다. 이는 매뉴얼 개발에서도 예외가 아니다.

이 문제에 대한 가장 이상적인 해법이라고 할 수 있는, 싱글 소스 퍼블리싱 single-source publishing은 —동일한 내용을 서로 다른 매체나 형식으로 만드는 것도 가리키지만— 여러 정보 덩어리들을 선택적으로 조합하여 하나의 문서로 만드는 것을 의미한다. 이 절은 싱글 소스 퍼블리싱을 구현할 때 고려해야 할 점들을 다룬다.

조건식

\NewConditionals 명령을 이용하여 조건식 명령과 환경을 만들 수 있다. 이것이 싱글 소스 퍼블리싱을 가능하게 한다.¹

```
\NewConditionals{명령}{설정}[환경]
```

아래와 같은 선언에 의해 \ifdoc과 \DocumentSetup 명령이, 그리고 IfDoc 환경이 생성되어 있다.

```
\NewConditionals{doc}{Document}[Doc]

\ifdoc*^{조건}[추가 조건]{ TRUE }[ FALSE ]
\DocumentSetup*{조건, 조건, ...}
\begin*^{IfDoc}{조건}[추가 조건] \end{IfDoc}
```

먼저 \DocumentSetup 명령을 이용하여 조건들을 저장해야 한다. \DocumentSetup은 주어진 조건들을 추가로 저장하는 반면, \DocumentSetup*은 앞서 저장된 것들을 덮어쓴다. \ifdoc 명령과 IfDoc 환경의 용법은 다음과 같다.

```
\DocumentSetup{foo, goo, ...}

\ifdoc{foo}{TRUE}[FALSE]      : foo가 목록에 있다면 TRUE 아니면 FALSE
\ifdoc^{foo}{TRUE}[FALSE]     : foo가 있지 않다면
\ifdoc{foo}[goo]{TRUE}[FALSE] : foo 또는 goo가 있다면
\ifdoc*{foo}[goo]{TRUE}[FALSE] : foo와 goo가 있다면

\begin{IfDoc}{foo} \end{IfDoc}
\begin{IfDoc}^{foo} \end{IfDoc}
\begin{IfDoc}{foo}[goo] \end{IfDoc}
\begin{IfDoc}*{foo}[goo] \end{IfDoc}
```

별표(*)와 caret(^)를 동시에 쓰는 것은 허용되지 않는다.

```
\DocumentSetup{iPhone}

\begin{IfDoc}{iPhone}
이 제품은 아이폰을 지원합니다.
\end{IfDoc}
\begin{IfDoc}{Android}
이 제품은 안드로이드 스마트폰과 호환됩니다.
\end{IfDoc}
```

이 제품은 아이폰을 지원합니다.

싱글 소싱 퍼블리싱을 위해 다음과 같은 조건식 처리 장치들을 추가로 만들 수 있을 것이다.

¹https://youtu.be/1w1Q-Kn3_g4을 보시라. 이 영상이 실제 사례를 보여준다.

```

\NewConditionals{spec}{Specification}[Spec]

\ifspec*{ }{ }{ }{ }
\SpecificationSetup{ }
\begin*{IfSpec}{ }{ } \end{IfDoc}

```

일부 항목을 별개 변수로 다루어야 한다면, 생성된 설정 명령을 이용하는 새로운 명령을 정의하라.

```

\keys_define:nn { modelspec }
{
  modelid .tl_set:N = \modelid,
  spec .code:n = { \SpecificationSetup{#1} }
}

\NewDocumentCommand\SpecSetup { m }
{
  \keys_set:nn { modelspec }{ #1 }
  \exp_args:No \SpecificationSetup{ \modelid }
}

```

이미지 경로

한 기업에서 제공하는 프로젝터 제품들을 위한 매뉴얼을 만든다고 가정하자. 이 제품들은 다양한 언어의 사용자 인터페이스를 제공한다. 지원하는 입출력 장치들의 구성이 모델마다 조금씩 다르고 그에 따라 메뉴 화면이 달라진다.

당장 맞닥뜨리는 문제는, 스크린샷 파일들에 저마다 다른 이름을 붙일 것인가 아니면 같은 이름을 붙일 것인가? 후자를 선택할 수밖에 없다. 왜냐하면 많은 이미지 파일들이 고유한 이름을 갖게 되면 그것들을 선택하기 위한 조건식이 감당하기 어려울 만큼 매우 복잡해지기 때문이다.

```

\ifspec*[korean][feature1]{ \image{korean_feature1_menu.jpg} }
\ifspec*[korean][feature2]{ \image{korean_feature2_menu.jpg} }
\ifspec*[english][feature1]{ \image{english_feature1_menu.jpg} }
\ifspec*[english][feature2]{ \image{english_feature2_menu.jpg} }

```

같은 용도의 스크린샷들에 동일한 파일 이름이 사용된다면 기능이나 언어에 따라 여러 폴더에 나누어 담아야 할 수밖에 없다.

```

+---images/
+---screenshot/
+---korean/
+---feature1/
+---menu.jpg
+---settings.jpg
+---feature2/
+---menu.jpg
+---settings.jpg
+---english/
+---feature1/

```

```

        +---menu.jpg
        +---settings.jpg
+---remotecontrol/
+---type1/
    +---button_menu.jpg
+---type2/
    +---button_menu.jpg

```

이제 문제는 이미지 경로를 올바르게 잡는 것이다. 다행히 `graphicx` 패키지가 복수의 이미지 경로를 지원한다.

```
\graphicspath{ {images/}{../images/}{../images/xxx/} }
```

이제 저 프로젝터 매뉴얼을 위해 이미지 경로를 설정하는 매크로를 만들어 보자.

```

\ExplSyntaxOn
\keys_define:nn { imagepath }
{
    root          .tl_set:N = \l_ipath_root_tl,
    language      .tl_set:N = \l_ipath_language_tl,
    feature       .tl_set:N = \l_ipath_feature_tl,
    remotecontrol .tl_set:N = \l_ipath_remote_tl
}
\NewDocumentCommand \ImagePathSetup { m }
{
    \keys_set:nn { imagepath } { #1 }
    \tl_set:Nn \l_tmpa_tl {
        {\l_ipath_root_tl/\l_ipath_language_tl/\l_ipath_feature_tl/}
    }
    \tl_put_right:Nn \l_tmpa_tl {
        {\l_ipath_root_tl/\l_ipath_remote_tl/}
    }
    \exp_args:Nx \graphicspath { \l_tmpa_tl }
}
\ImagePathSetup{
    root=../images,
    language=korean,
    feature=feature1,
    remotecontrol=type2
}
\ExplSyntaxOff
\ShowGraphicspath[\small]

```

| {../images/korean/feature1/}{../images/type2/}

`\ShowGraphicspath` 명령을 이용하여 이미지 경로가 올바르게 설정되었는지 확인할 수 있다.

이미지들이 고유한 파일 이름을 갖는다면 아이디를 이용하는 것이 한 가지 방법이 될

수 있다.

```
\keys_define:nn { ID }
{
  language      .tl_set:N = \languageid,
  feature       .tl_set:N = \featureid,
  remotecontrol .tl_set:N = \remoteid
}
\NewDocumentCommand \IDsetup { m }
{
  \keys_set:nn { ID } { #1 }
}
\IDsetup{
  language=korean,
  feature=feature1,
  remotecontrol=type2
}
\image{\languageid\_featureid\_menu.jpg}
\image{\remoteid\_button\_menu.jpg}
```

텍 파일 경로

텍 파일들이 여러 언어로 번역된다면 이미지 파일들과 마찬가지로 고유한 파일 이름을 붙일지 고민하게 된다. 하지만 이것은 별 문제가 되지 않는다. 어떤 방법을 쓰든, 이미지와 달리 언어라는 한 가지 조건밖에 없어서 조건식이 간단하기 때문이다. 각 파일에 고유한 이름을 붙인다면 localtexmf나 texmfhome 폴더에 파일들을 둘 수 있다. 이것은 텍 라이브의 파일 데이터베이스를 이용한다는 것인데, 경로 없이 이름만 지정하여 파일을 불러오는 것이 가능하다는 것을 뜻한다.

```
+---common/
+---korean/
+---introduction_kor.tex
+---installation_kor.tex
+---english/
+---introduction_eng.tex
+---installation_eng.tex
```

경로를 지정하여 파일을 불러오는 방법을 쓰겠다면 다음 예와 같은 명령들을 만들어야 할 것이다.

```
\keys_define:nn { texpath }
{
  root      .tl_set:N = \g_tex_path_root,
  language  .tl_set:N = \g_tex_path_language
}
\NewDocumentCommand \TexPathSetup { m }
{
  \keys_set:nn { texpath } { #1 }
}
\NewDocumentCommand \pathinput { s m }
{
```

```

\IfBooleanTF { #1 }
{ \input{\g_tex_path_root/#2} }
{ \input{\g_tex_path_root/\g_tex_path_language/#2} }
}
\TeXPathSetup{
  root=../common,
  language=korean
}

```

파일 외부에서 설정 변경하기

아래 예와 같이, 이러저러한 이유로 독자에 따라 여러 설정들을 자주 바꾸어야 한다면 레이텍 컴파일이 성가신 일이 된다.

```

\ImageSetup{scale=0.75, showfilename=true}
\ImageSetup{scale=1, showfilename=false}

```

그런 수고를 덜고자 다음과 같은 장치들이 고안되었다.

```

\FinalizerSetup{draft={...}, final={...}}
\FinalierOff
\FinalizerOn

```

\FinalizerSetup 명령으로 여러 설정들을 두 무리로 묶어야 한다.

```

draft      = \ShowImageFilename*\ShowPending ...
            \FinalizerOff 명령이 선언되면 이 옵션에 설정된 것들이 유효해진다.

final      = \HidePending ...
            \FinalizerOn 명령이 선언되면 이 옵션에 설정된 것들이 유효해진다.

```

이제 \FinalizerOff를 \FinalizerOn으로, 또는 그 반대로 고치는 것만으로도 일이 한결 수월해졌다. 아래 예처럼 파워셸 스크립트나 다른 유사한 방법을 사용하면 이 불편을 완전히 해소할 수 있다.

```

$tex="foo.tex"
if ($args[0] -eq "-d") {
  ((get-content $tex -raw) -replace "\\FinalizerOn", "\\FinalizerOff") |
  ↪ set-content $tex
}
if ($args[0] -eq "-f") {
  ((get-content $tex -raw) -replace "\\FinalizerOff", "\\FinalizerOn") |
  ↪ set-content $tex
}
xelatex $tex

```


제 7 장

소스 코드

이 문서를 작성하기 위하여 레이텍 코드를 처리하는 몇 가지 명령들과 환경들이 고안되었다. 그것들은 레이텍 설명서뿐만 아니라 프로그래밍 언어나 명령행에서의 입출력을 예시하는 데에도 요긴할 것이다.

7.1 레이텍 명령

편의를 위해 중괄호 braces 를 쓰도록 `\verb{...}` 명령이 재정의되었다.

```
\macro{\foo} \macro{foo}
\macro[색인 위치]{/foo}
\macro*{\foo} \macro*{foo}
\begin{macros}*!{옵션}(표제어 폭)<상위 색인> \item[] \end{macros}
```

`\macro` 명령이 레이텍 명령들이나 그것들의 옵션을, `\IndexingEnable` 명령이 선언되어 있다면 색인에 넣고, 고정폭 폰트로 식자한다. 백슬래시가 아닌 다른 기호로 시작하는 문자열에 대해 색인 위치를 옵션 인자로 지정할 수 있다. 그러나 별표가 붙은 `\macro*` 명령은 인자를 색인에 넣지 않는다. `\NewTerms`에 의해 만들어진 `macros` 환경은 레이텍 명령들이나 그것들의 옵션들을 표제어로 허용하는 목록 환경이다. 표제어가 두 단어 이상으로 이루어져 있다면 `\item[{}]` 또는 `\vitem{}`을 이용하라.

공백을 포함하는 표제어들은 색인에 포함되지 않는다. 색인 옵션에 대해서는 47 페이지 **색인**을 보라.

7.2 코드 예시하기

```
\begin{code} \end{code}
\begin{coderesult} \end{coderesult}
\CodeSetup{옵션}
```

`code` 환경이 소스 코드를, `coderesult` 환경이 소스 코드와 함께 그 결과를 함께 보여준다. `\CodeSetup` 명령으로 이 환경들의 설정을 변경할 수 있다.

```
before skip = .25\onelineskip
```

소스 코드 위 간격

afterskip = .25\onelineskip
소스 코드 아래 간격

font = \ttfamily
소스 코드의 폰트

fontsize = \small
폰트 크기

linespacing = 1.1
줄 간격

width = 0pt
박스의 폭. 0 포인트가 지정되면 현재 글줄의 길이가 사용된다.

offset = -\onelineskip
박스 안에서 위 테두리와 첫 줄 사이의 간격

language = latex
구문 강조 `syntax highlighting` 를 위해 적용할 언어. `minted` 클래스 옵션이 주어진 경우에 이 옵션이 유효하다.

codebox = FrameBox, LeftBarBox, RightBarBox, ShadeBox, ShadowBox, RuleBox, TabBox
소스 코드를 보여주는 박스의 모양. 51 페이지 [글상자](#)를 보라.

resultbox = FrameBox, ...
결과를 보여주는 박스의 모양

file = \jobname.vrb
소스 코드를 임시로 저장하는 파일의 이름

vbar = 옵션
이것은 기본 설정과 다른 작동을 허용하기 위한 목적으로 고안되었다. 주어진 옵션이 수직선(!)이 붙은 `\coderead!`에 적용된다. 디폴트는 `language=text` 이다.

`\verb` 명령과 `verbatim` 환경은 다른 명령이나 환경에 인자가 될 수 없기 때문에, 소스 코드의 저장과 읽기를 분리해야 하는 경우들이 있다.

```
\begin{codewrite} \end{codewrite}
\coderead*[옵션]
\codeinput
\begin{codedesc} \end{codedesc}
```

`codewrite` 환경은 소스 코드를 임시 파일에 저장한다. `\coderead` 명령은 저장된 소스 코드를 읽고 지정된 모양의 박스 안에 식자하는 반면, 별표가 붙은 `\coderead*`는 박스를 사용하지 않는다. `\codeinput` 명령은 저장된 소스 코드를 텍 파일로서 삽입한다. `codedesc` 환경은 저장된 소스 코드를 읽어서 왼쪽에 두고 오른쪽에 그에 대한 설명을 추가한다.

```
\begin{codewrite}
WIFI SSID dB|S:menu
xxxxxxx, -50dB
\end{codewrite}

\begin{codedesc}
연결된 와이파이 네트워크의 이름과 신호 세기
\end{codedesc}
```

```
WIFI SSID dB|S:menu
xxxxxxx, -50dB
```

연결된 와이파이 네트워크의 이름과 신호 세기

액정 표시 장치 같은 디스플레이에 표시되는 것들을 설명할 때 `codedesc` 환경을 직접 사용하기보다 다음 예와 같이 환경을 정의하여 사용하는 것이 좋다.

```
\newfontfamily\sgfamily{Segment7}
\NewDocumentEnvironment{segmentdesc}{}
{
  \begin{codedesc}[
    font=\sgfamily,
    fontsize=\normalsize,
    offset=-1\baselineskip,
    width=.11\textwidth
  ]
}{
  \end{codedesc}
}
```

7.3 공백과 빈 줄이 유지되는 환경

`lyrics` 환경에서 연속되는 공백과 빈 줄이 무시되지 않는다. 레이텍 명령을 사용할 수 있다는 점에서 `verbatim` 환경과 다르다.

```
\begin{lyrics}\ttfamily
Unicode code points : \textcolor{brown}{0xC720} 11000111 00100000

UTF-8 bytes      : \textcolor{violet}{EC9CA0} 11101100 10011100 10100000
\end{lyrics}
```

```
Unicode code points : 0xC720 11000111 00100000
```

```
UTF-8 bytes      : EC9CA0 11101100 10011100 10100000
```

소프트웨어 참조 설명서를 만드는 경우에 이 환경이 유용할 수 있다. 물론 시나 노랫말을 나타내는 데에도 쓸 수 있다.

제 8 장

문서 표지

ISO/IEC Guide 37:2012에 따르면 설명서는 제조사 이름, 주소, 웹 사이트를 비롯한 제조사 정보, 설명서가 적용되는 제품 유형과 모델을 비롯한 제품 정보, 설명서의 성격과 발행 날짜를 비롯한 문서 정보를 제공해야 한다. 이 요건들을 충족하는 문서 표지를 \FrontCover와 \BackCover 명령이 생성한다.

```
\CoverSetup{옵션}  
\FrontCover  
\BackCover
```

\CoverSetup 명령을 사용하여 아래 옵션들을 먼저 설정해야 한다.

`topskip` = `0.1\textheight`
 앞 표지에서 상단으로부터 띄우는 거리

`FrontLogoImage` = `foo.jpg`
 앞 표지에 넣을 로고 이미지

`BackLogoImage` = `foo.jpg`
 뒤 표지에 넣을 로고 이미지

`ProductImage` = `foo.jpg`
 제품 이미지

`FrontImage` = `foo.jpg`
 앞 표지를 위한 배경 이미지. 이미지를 놓는 기준점은 페이지 중앙이며, 페이지와 같은 크기의 이미지를 사용하는 것이 바람직하다.

`BackImage` = `foo.jpg`
 뒤 표지를 위한 배경 이미지

`title` = 문서 제목
 “SM-G998” 같은 제품 모델

`subtitle` = 부제
 “갤럭시 S21” 같은 제품 그룹 또는 유형.

TitleAlign = \raggedright
제목 정렬

DocumentType = 문서 유형
“User Guide”나 “설치 설명서” 같은 문서의 목적 또는 대상 독자

PubYear = 문서의 발행 년도
달리 지정하지 않으면 현재 년도

revision = 개정 번호
이것은 일반 출판물의 판 版과 같다. 제품의 수명 주기가 수년 이상일 때 그래서 여러 차례에 걸쳐 제품이 개선될 때 개정 번호가 유용할 수 있다. 소프트웨어 제품의 경우에 그 버전을 나타내는 데에 이 옵션을 사용할 수도 있을 것이다.

note = 문서에 대한 안내문
가장 흔히 사용되는 문구는 “언제든지 필요할 때 볼 수 있도록 이 문서를 가까운 곳에 보관하십시오” 또는 “나중에 참조할 수 있도록 이 매뉴얼을 잘 보관하십시오”이다.

manufacturer = 제조사의 공식 명칭
“㈜선경” 또는 “Apple Inc.”

address = 제조사 주소
본사 주소

AfterFront = \clearpage, \cleartooddpage
앞 표지 뒤에 쪽 나눔

BeforeBack = \clearpage, \cleartoevenpage
뒤 표지 앞에 쪽 나눔

FontI = \sffamily\bfseris
제목과 부제를 위한 폰트

FontII = \sffamily\bfseris
제목과 부제를 제외한 나머지에 적용되는 폰트

FontSizeI = \Huge
제목 크기

FontSizeII = \huge
부제 크기

FontSizeIII = \LARGE
문서 유형의 크기

FontSizeIV = \Large
발행 년도와 개정 번호의 크기

BlankFront = true/false
이 옵션이 true로 설정되면, 배경 이미지를 제외하고 앞 표지에 아무 것도

식자되지 않는다. 모든 문서 정보를 포함하는 이미지를 표지로 사용할 때 이 옵션이 유용하다.

```
\CoverSetup{
  FrontImage=foo.jpg,
  BlankFront=true
}
```

BlankBack 이 옵션이 true로 설정되면, 배경 이미지를 제외하고 뒤 표지에 아무 것도 식자되지 않는다.

hook = \foo

이 옵션을 이용하여 모서리에 장식을 추가하는 것과 같은 명령을 삽입할 수 있다.

```
\usepackage[object=vectorian]{pgfornament}
\colorlet{OrnamentColor}{Maroon!60}
\NewDocumentCommand \CoverOrnament { 0{61} }
{
  \begin{tikzpicture}[every node/.style={inner sep=0pt}, remember
  ⇨ picture, overlay]
  \node[shift={(.5cm, -.5cm)}, anchor=north west, color=OrnamentColor]
  ⇨ at (current page.north west) {\pgfornament[width=2cm]{#1}};
  \node[shift={(-.5cm, -.5cm)}, anchor=north east, color=OrnamentColor]
  ⇨ at (current page.north east) {\pgfornament[width=2cm,
  ⇨ symmetry=v]{#1}};
  \node[shift={(.5cm, .5cm)}, anchor=south west, color=OrnamentColor]
  ⇨ at (current page.south west) {\pgfornament[width=2cm,
  ⇨ symmetry=h]{#1}};
  \node[shift={(-.5cm, .5cm)}, anchor=south east, color=OrnamentColor]
  ⇨ at (current page.south east) {\pgfornament[width=2cm,
  ⇨ symmetry=c]{#1}};
  \end{tikzpicture}
}
\CoverSetup{hook=\CoverOrnament}
```


제 9 장

문서 템플릿

설명서들은 대개, 소프트웨어 문서들은 크게 다르지만, 비슷한 구조로 이루어져 있다. 소개-안전-설치-작동-유지 관리-문제 해결-보증. 이와 같은 열개를 가진 초고가 테크니컬 라이터가 반드시 기술해야 할 사항들을 잊지 않게 하는 데에 약간의 도움이 될 것이다.

`template` 클래스 옵션은 `hztemplate.tex`을 불러들인다. 그 파일에 정의되어 있는 여러 명령들을 조합하여 설명서 템플릿을 만들 수 있다.

`\tpltext` 한 문장이 식자된다. 선택적 인자에 최대 4가지 문장의 수를 지정할 수 있다.

`\tplpara` 두 문장이 식자된다. 별표가 붙은 `\tplpara*`는 세 문장을 식자한다.

`\tpllist` 세 항목이 포함된 `itemize` 목록이 만들어진다. 별표가 붙은 `\tpllist`는 `enumerate` 목록을 만든다. 또한, `\tpllist[terms][itemize]`와 같이, 선택적 인자에 환경 이름을 지정하여 중첩 목록을 만들 수 있다.

`\tplimage` `uncertain.pdf` 이미지가 놓인다.

`\tplimagetable` `ImageTable` 환경으로 만들어진 표가 놓인다.

`\tplprocedure` `IllustEnum` 환경으로 만들어진 절차가 놓인다. 선택적 인자에 단계의 수를 지정할 수 있다. 디폴트는 3이다.

`\tpltopics` 순서 목록을 포함하는 절이 만들어진다. 선택적 인자에 절의 수를 지정할 수 있다. 디폴트는 3이다.

`\tplspectables` `SpecTable` 환경으로 만들어진 표들이 놓인다. 선택적 인자에 표의 수를 지정할 수 있다. 디폴트는 3이다.

`\tpltable` 표 하나가 놓인다.

`\tplproblems` 하위 절과 순서 목록을 포함하는 절이 만들어진다. 선택적 인자에 절의 수를 지정할 수 있다. 디폴트는 2이다.

```
\tpllist[terms][itemize]
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante.

Term A: Lorem ipsum dolor sit amet, consectetur adipiscing elit.

- Lorem ipsum dolor sit amet, consectetur adipiscing elit.
- Lorem ipsum dolor sit amet, consectetur adipiscing elit.
- Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Term B: Lorem ipsum dolor sit amet, consectetur adipiscing elit.

- Lorem ipsum dolor sit amet, consectetur adipiscing elit.
- Lorem ipsum dolor sit amet, consectetur adipiscing elit.
- Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Term C: Lorem ipsum dolor sit amet, consectetur adipiscing elit.

- Lorem ipsum dolor sit amet, consectetur adipiscing elit.
- Lorem ipsum dolor sit amet, consectetur adipiscing elit.
- Lorem ipsum dolor sit amet, consectetur adipiscing elit.

문서 템플릿의 일부가 다음과 같이 이루어질 수 있다.

```
\chapter{Introduction} \tplpara
\section{Features} \tpllist
\section{Package Items} \tplimagetable
\section{Specifications} \tplspectables
```

부록 A

파이선 스크립트

<https://github.com/YiHoze/texwrapper>

```
C:\>python c:\home\bin\foo.py optional_argument mandatory_argument
C:\>foo opt_arg man_arg
```

위 줄이 아니라 아래 줄과 같은 방법으로 윈도우에서 파이선 스크립트를 실행할 수 있게 하려면 다음과 같이 설정해야 한다.

- PATHEXT 환경 변수에 .py를 등록한다. .py 파일들이 실행 가능한 것들로 인식된다.
- CPython에는 py.exe python launcher가 포함되어 있고, py.exe가 설치되면 별다른 설정 없이도 .py 파일들이 py.exe와 연결된다. 아나콘다 Anaconda에는 py.exe가 포함되어 있지 않아서 별도로 설치해야 한다. 권장되는 방법이 아니지만, python.exe와 연결하려면 다음과 같이 한다.

```
C:\>cmd /c assoc .py
C:\>cmd /c ftype Python.File="C:\Users\...\python.exe" %1 %*
```

- .py 파일들을 PATH 환경 변수에 포함된 경로에 둔다.

레이텍 사용을 비롯하여 문서 제작에 수반되는 여러 일들을 수월하게 하고자 다음과 같은 여러 래퍼 wrapper 스크립트가 파이선으로 작성되었다.

ltx.py lualatex 또는 xelatex을 이용하여 텍 파일을 컴파일한다.

i.py 현재 폴더에서 텍 파일을 찾아 ltx.py를 이용하여 컴파일한다.

iu.py 다양한 포맷의 이미지 파일들을 다른 포맷으로 변환한다.

fontinfo.py 설치된 폰트들 또는 특정 폰트의 정보를 보여준다.

tlconf.py 텍 라이브를 위한 환경을 설정한다.

op.py 텍스트 파일들과 PDF 파일들을 미리 지정된 프로그램으로 연다.

worddig.py 특정 문자열을 찾아 다른 문자열로 바꾼다.

unit.py 특정 단위의 값을 다른 단위로 변환한다.

fu.py 파일들을 백업한다.

이들 스크립트의 대부분이 공통으로 참조하는 설정 파일인 docenv.conf는 다음과 같은 섹션들을 포함하고 있다.

```
[LaTeX]
compiler = xelatex.exe

[TeX Live]
texmflocal = C:\texlive\texmf-local\tex\latex\local

[SumatraPDF]
path = C:\Program Files\SumatraPDF\SumatraPDF.exe
```

A.1 ltx.py: 텍 파일 컴파일하기

ltx.py는 latexmk.exe와 비슷한 레이텍 래퍼 wrapper 이다. 텍스트 에디터에서 단축키를 이용하여 텍 파일을 컴파일할 수 있지만, -shell-escape 옵션을 주거나 xindex 또는 bibtex을 돌려야 하는 것과 같은 다양한 경우들의 각각에 매우 제한적으로 서로 다른 단축키를 할당할 수 있을 뿐이다. 명령행 사용이 거북하지 않더라도, 이렇게 명령하는 것은 다소 번거롭다.

```
C:\>latexmk -lualatex -shell-escape foo.tex
```

latexmk를 사용하지 않은 가장 큰 이유는 초보자 시절에 그것의 사용법이 거추장스러워 보였고 그래서 자연스럽게 그것에서 멀어졌다.

```
C:\>ltx -L -s foo.tex
```

- L docenv.conf에 설정된 디폴트 컴파일러가 무시되고, lualatex이 실행된다. docenv.conf가 존재하지 않으면, xelatex이 디폴트 컴파일러로 설정된다.
- X xelatex이 실행된다.
- A docenv.conf의 alternative_path 옵션에 설정된 폴더에 있는 컴파일러가 사용된다. 예를 들어, PATH 시스템 변수에 C:\texlive\2022\bin\win64를 설정하고, alternative_path 옵션에 C:\texlive\2022\bin\win32를 설정하라.
- b 이것은 -interaction=batchmode를 의미한다. 이 옵션이 주어지지 않으면 -synctex=1이 사용된다.
- s 이것은 -shell-escape와 같다.
- w 주어진 텍 파일이 두 번 컴파일된다.
- f 세 번 컴파일된다. .idx 파일이 있으면 처리된다.
- v 생성된 PDF 파일이 op.py에 의해 열린다.

- n 컴파일이 생략된다. xindex나 bibtex 같은 다른 프로그램만 실행하려 할 때 이 옵션이 유용하다.
- i 색인이 정렬된다.
- I 색인 정렬을 위한 언어를, `german` 또는 `ger`와 같이, 지정하라. 디폴트는 `korean`이다. `.ist` 또는 `.xdy`를 지정하는 것도 가능하다.
- a -f 옵션이 주어지면 컴파일 뒤에 `.aux`를 비롯한 모든 부수 파일들이 삭제된다. 그 파일들을 남겨두려면 이 옵션을 사용하라.
- m 모든 색인 항목들이 PDF 북마크에 추가된다.
- M 이것은 파이썬 독스트링 `docstring` 을 위한 것이다. 색인 항목들 가운데 파이썬 함수들이 PDF 북마크에 추가된다.
- c 모든 부수 파일들이 삭제된다.
- B bibtex이 실행된다.
- P pythontex이 실행된다.

색인 정렬 프로그램을 선택하는 과정은 다소 복잡하다.

```
C:\>ltx -i -I eng foo
C:\>ltx -i -I foo.xdy foo
C:\>ltx -i -I kotex.ist foo
```

지정된 언어가 미리 정의된 xindex 언어 목록에 포함되어 있으면 xindex가 사용된다. 없다면 그 언어가 xindy 언어 목록에 있는지 확인하고, 있으면 texindy가, 없으면 xindex가 영어를 기준으로 `.idx`를 처리한다. `.xdy` 파일이 지정되면 texindy가 사용되고, `.ist`가 지정되면 komkindex가 사용된다.

참고

xindex를 사용하여 한글 색인을 정렬하는 데에 `xindex-hz-ko.lua`가 사용되고, 그것은 `xindex-ko.lua`를 부른다.

참고

레이텍 명령들 때문에 이 문서의 색인을 정렬하는 데에 `hzguide.xdy`가 사용되었다.

A.2 i.py: 더 게으른 사용자를 위한 레이텍 래퍼

`i.py`는 `ltx.py`를 사용하는 레이텍 래퍼이다. 이 스크립트는 다음과 같이 작동한다.

1. C:\>i

현재 폴더에서 `i.ini` 파일을 찾는다. 없다면 `.tex` 파일들을 찾는다. 발견된 파일이 하나이면 그 파일의 이름을 아래와 같이 `i.ini` 파일에 기록하고 컴파일한다.

```
[tex]
target = foo.tex
```

2. 둘 이상의 텍 파일들이 있다면, 번호와 함께 파일 이름들이 나열된다. 번호를 입력하여 컴파일할 파일을 선택하라. “2 5 8-10”과 같이 복수의 파일들을 선택할 수 있다. 선택된 파일들의 이름이 i.ini에 기록되고 컴파일된다. 0을 입력하면 모든 파일들이 i.ini에 기록되고 컴파일된다.
3. i.ini 파일에 기록된 텍 파일이 하나이면 그것이 컴파일된다. 둘 이상이면, 사용자가 선택할 수 있도록 번호와 함께 파일 이름들이 나열된다. “2 5 8-10”과 같이 복수의 파일들을 선택할 수 있다. 0을 입력하면 모든 파일들이 컴파일된다.
4. C:\>i foo
현재 폴더에 i.ini 파일이 없다면, 파일 이름에 foo를 포함하는 텍 파일들을 찾는다. 나머지 과정은 첫째 및 둘째 단계와 동일하다.
5. i.ini 파일이 있고, 기록된 파일 이름들 가운데 foo를 포함하는 것이 하나이면 그것이 컴파일된다. 나머지 과정은 셋째 단계와 동일하다.

i.ini는, 다양한 방법으로 처리할 수 있도록, 여러 옵션들을 갖고 있다.

```
[tex]
target = foo.tex
compiler =
draft = wordig.py -a "..." -s "..." %(target)s
final = wordig.py -a "..." -s "..." %(target)s
final_compiler =
after =
main = \input{preamble}
      \begin{document}
      \maketitle
      \input{\1}
      \end{document}
```

i.py를 다음과 같은 선택 옵션들과 함께 사용할 수 있다. i.py가 인식하지 못하는 옵션들은 ltx.py에 전달된다. i.ini의 compiler에 설정된 것들도 ltx.py에 전달된다.

- U i.ini를 업데이트할 수 있도록 텍 파일들을 나열한다.
- D 초본을 만들기 위한 전처리로서 i.ini의 draft에 지정된 명령이 수행된다.
- F 최종본을 만들기 위한 전처리로서 i.ini의 final에 지정된 명령이 수행된다. 그리고 final_compiler에 설정된 옵션들이 ltx.py에 전달된다.
- W 이것은 여러 하위 파일들을 따로 컴파일하기 위한 목적으로 고안되었다. i.ini의 main 옵션이 위 예와 같이 설정되어야 한다. \1이 선택된 파일로 바뀐다.
- N 컴파일이 생략된다.
- C i.ini 파일이 생성된다.

A.3 iu.py: 이미지를 다른 포맷으로 변환하기

iu.py에 정의된 클래스 이름이 ImageUtility인데, 그것은 억지로 지어 붙인 것이고, 이 파일 이름은 그림처럼 이쁜 아이유를 기리기 위한 것이다. 이것 역시 래퍼이기 때문에 다음과 같은 프로그램들이 필요하다.

- 이미지매직 ImageMagick
- 고스트스크립트 GhostScript
- 텍 라이브 epstopdf.exe, pdfcrop.exe, pdftops.exe
- 잉크스케이프 Inkscape

iu.py는 다음과 같은 것들을 할 수 있다.

- 비트맵 이미지들을 다른 포맷의 이미지로 변환할 수 있다.

```
C:\>iu -t png *.jpg
```

- 비트맵 이미지들을 PDF 또는 EPS로 변환할 수 있다. 정확히 말해, 이미지매직이 고스트스크립트에게 그 작업을 맡긴다.

```
C:\>iu -t pdf *.png
```

- 벡터 이미지들을 비트맵 또는 다른 포맷의 벡터 이미지로 변환할 수 있다.

```
C:\>iu -t png *.eps  
C:\>iu -t eps *.svg
```

- 비트맵 이미지들의 픽셀 크기나 해상도를 변경할 수 있다.

```
C:\>iu -r -s 75 *.jpg  
C:\>iu -r -d 150 *.jpg
```

- 비트맵 이미지들의, 픽셀 크기를 포함하는, 이미지 정보를 볼 수 있다.

```
C:\>iu -i foo.jpg
```

-t 이 옵션과 함께 지정된 포맷 eps, pdf, svg, bmp, cr2, gif, jpg, jpeg, pbm, png, ppm, tga, tiff, webp 으로 변환된다. 디폴트는 pdf이다. GIF 이미지와 여러 페이지로 이루어진 PDF 파일은 낱개 이미지로 쪼개진다.

-T 목표 포맷이 PNG인 경우에 흰색을 투명으로 바꾼다.

-r 화소 밀도를 조정하여 이미지의 인쇄 크기를 변경한다.

-d 해상도를 지정하라. 단위는 ppc pixels per centimeter 이고, 디폴트는 100이다.

-m 아래 예의 경우에 가로 800 픽셀보다 큰 이미지들이 800 픽셀로 축소된다.

```
C:\>iu -r -m 800 *.png
```

- s 75를 지정하면 원래 크기의 75 퍼센트로 바뀐다.
- R 모든 하위 폴더에 있는 이미지 파일들이 처리된다.
- I EPS를 PDF로 또는 그 반대로 변환하는 데에, `epstopdf.exe`와 `pdftops.exe` 대신, 잉크스케이프가 사용된다.
- c `pdfcrop.exe`가 PDF 이미지들에서 여백을 제거한다.

A.4 fontinfo.py: 폰트 정보 보기

`fontinfo.py`는 텍 라이브에 포함된 `fc-list.exe`와 `otfinfo.exe`를 이용하여 설치된 폰트들에 대한 정보를 보여준다. 사용 방법은 다음과 같다.

1. 텍 라이브를 포함하여 설치된 모든 폰트들의 목록이 `fonts_list.txt`에 만들어진다. 출력 파일의 이름을 바꾸려면 `-o` 옵션을 사용하라.

```
C:\>fontinfo
C:\>fontinfo -o myfonts.txt
```

2. 폰트 이름 또는 파일 이름을 지정하면, 언어별로 짧은 문장을 보여주는, 같은 이름의 PDF가 만들어진다. 이 기능은 `mytex.py`와 `latex.db`를 요구한다. 90 페이지 [레이텍 템플릿 모음](#)을 보라.

```
C:\>fontinfo "Noto Serif"
C:\>fontinfo NotoSerif-Regular.ttf
```

3. 폰트 정보를 보려면 `-i` 옵션과 함께 폰트 이름 또는 파일 이름을 지정하라.

```
C:\>fontinfo -i "Noto Serif"
C:\>fontinfo -i NotoSerif-Regular.ttf
```

A.5 tlconf.py: 텍 라이브와 관련된 설정

텍 라이브를 처음 설치한 다음에 설정해야 할 것들이 몇 가지 있다. `tlconf.py`는 텍 라이브를 위한 설정들을 수월하게 처리하고자 고안되었다. `docenv.conf` 설정 파일에서 다음 옵션들이 적절하게 설정되어 있어야 한다.

```
[TeX Live]
texmflocal = C:\texlive\texmf-local\tex\latex\local
repository_main = https://cran.asia/tex/systems/texlive/tlnet/
repository_private = http://ftp.ktug.org/KTUG/texlive/tlnet

TEXEDIT = code.exe -r -g %s:%d
TEXMFHOME = C:\home\texmf

[LOCAL.CONF]
path = C:\texlive\2021\texmf-var\fonts\conf\local.conf
```



```
content = <dir>C:/Users/.../AppData/Local/Microsoft/Windows/Fonts</dir>
```

[SumatraPDF]

```
path = C:\Program Files\SumatraPDF\SumatraPDF.exe
```

```
inverse-search = code.exe -r -g %f:%l
```

```
C:\>tlconf -b
```

- L local.conf 파일이 만들어지고 사용자의 폰트 폴더 경로가 그 파일에 추가된다. 텍 라이브 업데이트가 간혹 이 파일을 삭제할 수 있다.
- H TEXMFHOME 환경 변수가 만들어지고 설정 파일에 지정되어 있는 경로로 설정된다.
- e TEXTEDIT 환경 변수가 만들어진다.
- p Sumatra PDF의 경로와 inverse search 옵션이 설정된다.
- r tlmgr.exe가 사용할 텍 라이브 저장소가 지정된다.
- u 텍 라이브가 업데이트된다.
- c xelatex을 위해 fc-cache가 폰트들을 캐시한다. 폰트 캐시가 반드시 필요한 것이 아니다. 새로 설치되어, 아직 캐시되지 않은 폰트를 사용하는 경우에 텍 파일이 컴파일되면서 그 폰트의 캐시가 만들어진다. 드물지만 폰트 캐시가 손상되는 경우가 있다. 그 경우에 모든 폰트의 캐시를 새로 만드는 것이 해법이다.
- l lua_latex을 위해 luaotfload-tool이 폰트 데이터베이스를 갱신한다.
- f 모든 포맷 파일들이 새로 생성된다. 이것이 64비트 lua_latex이 정상적으로 작동하지 않을 때 해결책이 될 수 있다.
- b 이것은 위의 모든 옵션을 선택한 것과 동일하다.
- q 사용자 확인을 묻지 않고 작업들이 진행된다.

HzGuide 클래스와 함께 제공되는 tlconf.cmd는 tlconf.py의 간단 버전이다.

```
C:\>tlconf.cmd cnf
C:\>tlconf.cmd texedit
C:\>tlconf.cmd texmfhome
C:\>tlconf.cmd sumatrapdf
C:\>tlconf.cmd batch
```

아무 인자도 주지 않으면 각 옵션에 대한 도움말이 나온다.

cnf 사용자 폰트 폴더의 경로를 포함하는 local.conf를 생성한다.

texedit TEXTEDIT 환경 변수를 설정한다.

texmfhome TEXMFHOME 환경 변수를 설정한다.

sumatrapdf Sumatra PDF에 인버스 서치 `inverse search` 옵션을 설정한다.

batch 이것은 위의 모든 옵션들을 지시한 것과 동일하다.

```

@echo off

for /f "usebackq delims=" %%p in (`kpsewhich -var-value=TEXMFROOT`) do set tlroot=%%p
set tlroot=%tlroot:!=\%

if /i .%1. == .. goto help
if /i .%1. == .conf. goto conf
if /i .%1. == .texedit. goto texedit
if /i .%1. == .texmfhome. goto texmfhome
if /i .%1. == .sumatrapdf. goto SumatraPDF
if /i .%1. NEQ .batch. goto eof

:conf
REM checking if TeX Live is found
kpsewhich -var-value=TEXMFROOT >nul 2>&1
if errorlevel 1 (
echo TeX Live not on searchpath. Aborted.
exit /b
)

REM creating local.conf
set localconf=%tlroot%\texmf-var\fonts\conf\local.conf
set code=%LOCALAPPDATA%\Microsoft\Windows\Fonts
set code=%content:!=\%
set "code=^<dir^>%content%^</dir^>"
echo %content% > %localconf%
echo %localconf%:
type %localconf%
if /i .%1. NEQ .batch. goto eof

:texedit
if .%2. == .. (
setx TEXEDIT "code.exe -r -g %s:%d"
) else (
setx TEXEDIT "%2"
)
reg query HKEY_CURRENT_USER\Environment /v TEXEDIT
if /i .%1. NEQ .batch. goto eof

:texmfhome
if .%2. == .. (
setx TEXMFHOME "C:\home\texmf"
) else (
setx TEXMFHOME "%2"
)
reg query HKEY_CURRENT_USER\Environment /v TEXMFHOME
if /i .%1. NEQ .batch. goto eof

:SumatraPDF
reg query HKEY_LOCAL_MACHINE\SOFTWARE\Classes\Applications\SumatraPDF.exe\DefaultIcon >nul
↪ 2>&1
if errorlevel 1 (
echo SumatraPDF is not found.
goto eof
) else (
for /f "usebackq tokens=3-4 delims=" %%x in (`reg query
↪ HKEY_LOCAL_MACHINE\SOFTWARE\Classes\Applications\SumatraPDF.exe\DefaultIcon`) do set
↪ sumatra="%%x %%y"
)
set sumatra=start "" %sumatra% -inverse-search
if .%2. == .. (
%sumatra% "code.exe -r -g %%f:%%l"

```

```

) else (
%Sumatra% %2
)
goto eof

:help
set localconf=%tlroot%\texmf-var\fonts\conf\local.conf
echo %localconf%:
type %localconf%

echo.
echo.
echo tlconf.cmd conf          : Create local.conf with the user's local font directory.
echo tlconf.cmd texedit [...] : Set the TEXEDIT environment variable. The default is
echo                               code.exe -r -g %s:%d
echo tlconf.cmd texmfhome [...] : Set the TEXMFHOME environment variable. The default is
echo                               C:\home\texmf
echo tlconf.cmd sumatrapdf [...] : Set the inverse search command-line option of SumatraPDF.
↔ The default is
echo                               code.exe -r -g %f:%l
echo tlconf.cmd batch          : Proceed with all options.

:eof

```

A.6 op.py: 파일 열기

탐색기에서 어떤 파일을 열기 위해 특정 프로그램과 연결하는 일은 다소 성가시다. op.py는 docenv.conf에 지정된 프로그램들을 이용하여 파일들을 연다.

```

[SumatraPDF]
path = C:\Program Files\SumatraPDF\SumatraPDF.exe

[Text Editor]
path = C:\...\Microsoft VS Code\code.exe
associations = .aux, .bib, .bst, .cls, .cnf, .conf, .cmd, .css, .csv, .gv,
               .ini, .ind, .idx, .ist, .ipynb, .list, .lof, .log, .lot, .md, .ps1, .py,
               .rst, .sty, .tex, .tmp, .toc, .tsv, .txt

[Adobe Reader]
path = C:\Program Files (x86)\Adobe\Acrobat Reader DC\Reader\AcroRd32.exe

[Web Browser]
path = C:\Program Files (x86)\Google\Chrome\Application\chrome.exe

```

아래 예처럼 대상들을 다섯 가지로 나눌 수 있다.

```

C:\>op foo.tex
C:\>op foo.pdf
C:\>op -s memoir.cls
C:\>op -w ktug.org
C:\>op foo.docx

```

파일 이름의 일부만 지정할 수 있다. 그러면 파일 이름에 지정한 문자들이 포함된 모든 파일들이 열린다.

명시되지 않은 확장자의 파일을 여는 것은 윈도우가 결정한다.

-A PDF 파일을 여는 데에 아크로벳 리더가 사용된다.

-a 달리 사용할 응용 프로그램을 지정하라.

```
C:\>op -a notepad foo.txt
```

-o -a 옵션과 함께, 응용 프로그램에 전달할 옵션들을 지정하라.

-s 텍 라이브에 포함된 파일을 열려면 이 옵션을 사용하라.

-S 텍 라이브에 포함된 파일의 경로를 찾아서 클립보드에 복사한다.

-f 이미지 파일 같은 바이너리 파일을 포함하여, 지정되지 않은 형식의 파일을 텍스트 에디터로 열려면 이 옵션을 사용하라.

-w 웹 주소를 열려면 이 옵션을 사용하라.

A.7 wordig.py: 문자열을 찾아 바꾸기

wordig.py는, 단어 수 세기를 비롯하여, 레이텍 사용자가 텍스트 파일을 갖고 할 법한 많은 일들을 처리한다.

-a 찾을 문자열 또는 정규 표현식을 지정하라.

-A 한 줄에 하나씩, 찾을 문자열 또는 정규 표현식이 포함된 파일을 지정하라. 아래 변환 명세 파일의 규칙에 여기에 적용되지 않는다.

-s 바꿔치기할 문자열을 지정하라.

-P 변환 명세 파일을 지정하라. TSV —또는 CSV— 형식으로 작성된 파일에서 변환 명세 파일에서 찾기-바꾸기를 나타내는 한 쌍의 정규 표현식과 치환 문자열이 각 줄을 차지해야 한다. 변환 명세 파일에 다음과 같은 규칙이 적용된다.

- 두 개의 칼럼이 존재한다.
- 탭 또는 콤마로 칼럼이 구분된다.
- 첫 칼럼에 찾을 문자열이, 둘째 칼럼에 대체 문자열이 있다. 둘째 칼럼이 없거나 비어 있으면 찾을 문자열이 삭제된다.
- 빈 줄이 무시된다.
- ~~~로 시작하는 줄이 주석으로 간주된다.
- 정규 표현식에서 도트(.)는 개행 문자를 제외한 모든 문자에 대응된다. 주석 줄 끝에 DOTALL이 있으면, 그 다음 주석 줄을 만날 때까지 그 아래에 있는 모든 정규 표현식들의 처리에서 도트가 개행 문자를 포함하는 모든 문자에 대응된다.

-c 찾기에서 대소문자를 구분한다. 바꾸기를 위한 찾기에서는 항상 대소문자를 구분한다.

-L 이 옵션이 주어지면 도트가 개행 문자를 포함하는 모든 문자에 대응된다.

- e 주어진 정규 표현식에 일치하는 문자열들을 추출하여 파일별로 저장한다. -a 또는 -A 옵션이 설정되어야 한다.
- g 문자열들을 추출하고 한 파일에 합쳐 저장한다. 다음 예가 모든 텍 파일에서 단어들을 추출한다. 달리 지정하지 않으면 출력 파일의 이름이 gathered_strings.txt 이다.

```
C:\>wordig -a="\w+" -g *.tex
```

- o 출력 파일을 위한 이름을 지정하라. 디폴트는 기능에 따라 달라진다. 대상 파일의 이름이 foo.tex 일 때, 주어진 이름에 따라 출력 파일의 이름이 다음과 같이 결정된다.

```
-o "_replaced" → foo_replaced.tex
-o "replaced_" → replaced_foo.tex
-o ".txt" → foo.txt
-o "goo" → goo.tex
-o "goo.txt" → goo.txt
-o "sub\" → sub\foo.tex
-o "sub\_replaced" → sub\foo_replaced.tex
```

폴더 구분자 directory separator 로 백슬래시(\) 대신 슬래시(/)를 사용할 수 있다.

- v 출력 파일의 이름이 지정되지 않으면 --o 옵션이 주어지지 않으면— 문자열 치환이 그 결과를 원본 파일에 덮어쓴다. 이 옵션이 주어지면 같은 이름의 파일이 존재할 때 출력을 위한 파일 이름에 번호가 추가된다. 다른 기능이 수행될 때에도 마찬가지이다.
- r 모든 하위 폴더에 있는 파일들이 처리된다.
- p 각 PDF 파일들의 페이지 수와 합계를 구하려면 이 옵션을 사용하라.
- U 주어진 문자들의 유니코드 정보가 표시된다.

```
C:\>wordig -U 가나
```

- u 주어진 문자들의 유니코드 정보와 함께 UTF-8 비트들이 표시된다.
- X 주어진 16 진수 값의 유니코드 문자가 표시된다.

```
C:\>wordig -X AC00 B098
```

- D 주어진 10 진수 값의 유니코드 문자가 표시된다.
- E 대상 파일들이 지정된 인코딩에서 UTF-8로 변환된다. 달리 지정하지 않으면 UTF-8 폴더가 만들어지고 그 안에 출력 파일들이 저장된다.

```
C:\>wordig.py -E cp949 *.txt
```

- x .tsv 파일을 .xlsx로 변환한다.

-t .xlsx 파일을 .tsv로 변환한다.

아무 옵션도 지시하지 않으면 주어진 파일들의 글자 수와 단어 수가 표시된다.

```
C:\>wordig *.tex
C:\>wordig *.pdf
```

문자열을 찾거나 바꾸는 방법은 다음과 같다.

```
C:\>wordig -a "itemize" *.tex
C:\>wordig -a "network" *.pdf
C:\>wordig -a "itemize" -s "enumerate" *.tex
C:\>wordig -P foo.tsv *.tex
```

다음 예가 자동조사 매크로를 조사로 또는 반대로 바꾼다.

```
C:\>wordig -a="\([은는이가을를와과으로])" -s="\1" *.tex
C:\>wordig -a="(ref.+?\})([은는이가을를와과으로])" -s="\1\\2" *.tex
```

텍 매크로들을 제거하기 위한 목적으로 detex.tsv가 작성되었다.

```
(?<!\|\\)%.*?$
\\include.+$
\\begin\{.+\\}.+$
\\end\{.+\\}
\\[a-zA-Z*]{1,25}
```

```
C:\>wordig -P detex.tsv -o "_cleaned" *.tex
```

텍 매크로들을 추출하려면 다음과 같이 정규 표현식 파일을 작성한다.

```
\\[^\^a-zA-Z]
\\[a-zA-Z*^\^|+]+
\\begin(\\{.+?\\}[*^\^|+]*)
```

```
C:\>wordig -A extractex.txt -g *.tex
```

A.8 unit.py: 단위 변환하기

unit.py는 다음과 같은 여러 비표준 단위의 값들을 미터법 단위로 환산한다. 미국이 미터법을 채용하지 않은 거의 유일한 나라이다.

```
acre, degree, fahrenheit, foot, gallon, hp, inch, km/h, knot, mile, Newton,
↪ pascal, pint, point, pound, psi, pyeong, yard
```

다른 단위와 구분될 수 있을 만큼 사용할 단위의 처음 글자들을 숫자와 함께 지정하라. 아래 예에서 첫 줄의 p는 pascal로, 둘째 줄의 po는 point로 간주된다.

```
C:\>unit 10 p
C:\>unit 20 po
C:\>unit 30 pou
```

```
C:\>unit 30 py
```

천 자리 구분자로 쉼표를 사용할 수 있는데, 파워셸에서는 따옴표로 값을 감싸거나 이스케이프 문자인 역슬래시와 함께 쉼표를 써야 한다.

```
C:\>unit 2`,000`,000 mi
C:\>unit "2,000,000" mi
```

unit.py는 또한 RGB 색상을 CMYK로 또는 그 반대로 바꿀 수 있다. RGB는 HTML 색상 값이나 16 비트 10 진수로, CMYK는 백분율 값으로 지정해야 한다. 띄어쓰기를 하려면 값을 따옴표로 감싸야 한다.

```
C:\>unit -c 0000FF
C:\>unit -c 240,120,99
C:\>unit -c '240, 120, 99'
C:\>unit -c "0, 0.1, 0.33, 0.01"
```

10 진수의 16 진수를 또는 16 진수의 10 진수를 얻으려면 -x 옵션을 사용한다.

```
C:\>unit -x 44032
C:\>unit -x AC00
```

A.9 fu.py: 파일들 백업하기

fu.py는 파일들을 관리하기 위한 목적으로 고안되었다.

-d, --destination 목적 폴더를 지정하라. 디폴트가 기능에 따라 달라진다.

-r, --rename-files 파일 이름을 변경한다.

-a, --affix 파일 이름을 변경하기 위해 찾을 문자열 또는 정규 표현식을 지정하라.

-s, --substitute 파일 이름을 변경하기 위해 바꿔치기할 문자열을 지정하라.

-g, --gather-files 하위 폴더에 있는 것들을 포함하여, 파일들을 지정된 폴더로 복사한다.

-t, --total-size 하위 폴더에 있는 것들을 포함하여, 모든 파일들의 전체 크기를 구한다.

-f, --flag 각 기능에 사용할 방법을 지정하라.

파일들 백업하기

목적 폴더를 지정하지 않으면, _bak 폴더가 만들어지고, 그 아래에 파일들이 복사된다. -a나 --affix 옵션을 사용하여 파일들의 이름에 붙일 문자열을 지정할 수 있다. 달리 지정하지 않으면 오늘 날짜가 접사로 사용된다.

```
C:\>fu -d=c:\repository -a=adhoc *.tex *.pdf
```

같은 이름의 파일이 존재한다면 번호가 추가된다.

```
foo_2020-10-10.tex
foo_2020-10-10_1.tex
...
foo_2020-10-10_2.pdf
```

-f 또는 --flag 옵션을 사용하여 백업 방식을 지정할 수 있다.

0, file-today 파일 이름에 접사를 붙인다. 이것이 디폴트이다.

```
C:\>fu *.tex
_bak\foo_2020-10-10.tex
```

1, directory-today 접사와 같은 이름으로 하위 폴더를 만든다.

```
C:\>fu -f=1 *.tex
_bak\2020-10-10\foo.tex
_bak\2020-10-10_1\foo.tex
```

2, directory-file-today 접사와 같은 이름으로 하위 폴더를 만드고, 파일 이름에 접사를 붙인다.

```
C:\>fu --flag=directory-file-today *.tex
_bak\2020-10-10\foo_2020-10-10.tex
_bak\2020-10-10_1\foo_2020-10-10.tex
```

파일 이름 바꾸기

다음과 같이 파일 이름 변경을 위한 다양한 플래그 옵션들이 마련되어 있다.

0, append-letters 접사를 파일 이름 끝에 붙인다. 달리 지정하지 않으면 오늘 날짜가 접사로 사용된다.

```
C:\>fu -r *.pdf
foo.pdf -> foo_2020-10-10.pdf
```

1, prepend-letters 접사를 파일 이름 머리에 붙인다.

```
C:\>fu -r -f=1 *.pdf
foo.pdf -> 2020-10-10_foo.pdf
```

2, remove-letters 파일 이름에서 접사를 제거한다.

```
C:\>fu -r --flag=remove-letters --affix="_2020-10-10" *.pdf
foo_2020-10-10.pdf -> foo.pdf
```

3, replace-letters 파일 이름으로 접사를 지정된 문자열로 바꾼다.


```
C:\>fu -r --flag=replace-letters --affix="&" --suffix="#" *.pdf
foo&goo.pdf -> foo#goo.pdf
```

4, remove-spaces 파일 이름에서 공백을 제거한다.

```
C:\>fu -r -f=remove-spaces *.pdf
Korean User Guide.pdf -> KoreanUserGuide.pdf
```

5, uppercase 파일 이름에서 모든 로마자를 대문자로 바꾼다.

```
C:\>fu -r -f=uppercase *.pdf
user_guide.pdf -> USER_GUIDE.PDF
```

6, lowercase 파일 이름에서 모든 로마자를 소문자로 바꾼다.

```
C:\>fu -r -f=lowercase *.pdf
USER_GUIDE.PDF -> user_guide.pdf
```

7, ext-lowercase 파일 확장자만 소문자로 바꾼다.

```
C:\>fu -r -f=ext-lowercase *.png *.jpg
Image_1.PNG -> Image_1.png
Image_2.JPG -> Image_2.jpg
```

8, date-created 사진 파일의 경우에 파일 이름을 이미지가 생성된 (촬영된) 날짜로 바꾼다. 메타데이터가 없는 사진 파일들과 다른 포맷의 파일들에 대해서는 마지막 수정 날짜가 사용된다.

```
C:\>fu -r -f=date-created *.jpeg
IMG_1215.JPEG -> 2022-06-25_01.JPEG
IMG_1216.JPEG -> 2022-06-25_02.JPEG
```

A

파일들 모으기

그다지 생길 법한 일이 아니지만, 하위 폴더에 있는 것들을 포함하여, 특정 형식의 파일들을 한 군데로 모아야 할 때가 있다.

```
C:\>fu -g d=c:\repository c:\projects\*.pdf
```

달리 지정하지 않으면 현재 폴더로 파일들이 복사된다.

0, overwrite 같은 이름의 파일이 있으면 덮어쓴다.

1, append-number 같은 이름의 파일이 있으면 파일 이름에 번호를 붙인다.

```
C:\>fu -g -f=append-number *.pdf

.\foo.pdf
```

```
.\foo_1.pdf
```

전체 파일 크기 구하기

파일들이 복사되는 도중에 뜨는 공간이 부족하다는 메시지가 결코 유쾌하지 않다. 폴더를 복사하기 전에 파일들의 전체 크기를 알아내는 것이 현명하다.

```
C:\>fu -t c:\foo d:\goo  
  
c:\foo\ 12.34 MB  
d:\goo\ 5.6 GB
```

달리 지정하지 않으면, 현재 폴더 아래에 있는 모든 파일들의 크기를 합산한다.

A.10 trapper.py: 스크린샷 저장하기

Ctrl + **Shift** + **S** 키를 이용하여 스크린샷을 캡처하고 클립보드에 복사할 수 있다. 그러나 캡처한 이미지를 파일로 저장하려면 캡처하기 전에 “캡처 및 스케치”를 열거나 캡처한 뒤에 “그림판” 따위를 열어야 한다. 그 번거로움을 피하기 위해 trapper.py가 작성되었다. trapper.py는 클립보드에 있는 이미지를 파일로 저장한다.

```
C:\>trapper [foo]  
C:\>trapper -c foo.png
```

파일 이름을 지정하지 않으면, 2022-10-25_2와 같이 —같은 이름의 파일이 존재하면 번호를 추가하여— 오늘 날짜가 파일 이름이 된다. PNG 포맷으로만 저장된다. 클립보드에 있는 것이 이미지가 아니라 텍스트라면 그 텍스트가 출력된다. -c 옵션을 주면, 반대로, 지정된 이미지 파일이 클립보드에 복사된다.

A.11 레이텍 템플릿 모음

여러 레이텍 예들이 latex.db라는 데이터베이스 파일에 저장되어 있다. mytex.py가 latex.db에서 지정된 템플릿을 꺼내어 저장하고 컴파일한다. 다음은 latex.db의 일부이다.

```
[multilingual]  
description = Use this template to see how many languages are supported by a font.  
compiler = -c  
tex_output = multilingual.tex  
placeholders = 1  
defaults = Noto Serif  
tex = \documentclass{minimal}  
      \usepackage{fontspec}  
      \setlength\parskip{1.25\baselineskip}  
      \setlength\parindent{0pt}  
      \setmainfont{\1}  
      \begin{document}
```


- o 출력 파일의 이름을 지정하라.
- s placeholders 옵션이 지정된 템플릿에서 \1처럼 백슬래시로 시작하는 숫자가 이 옵션과 함께 주어진 문자열로 대체된다.

```
C:\>mytex -s "20, 10" lotto
```

- n compiler에 설정된 옵션들이 ltx.py에 전달되는데, -n가 주어지면 ltx.py가 호출되지 않는다.
- f compiler 옵션에 아무 것도 설정되어 있지 않아도 템플릿 파일이 ltx.py에 의해 컴파일된다.
- d PDF 파일을 제외하고 템플릿 파일을 비롯하여 부수적으로 생성된 모든 파일들이 컴파일 뒤에 삭제된다.
- l 템플릿들이 나열된다.
- L 각 템플릿의 설명과 함께 템플릿들이 나열된다.
- D 지정된 템플릿에 대한 상세한 설명이 표시된다.
- i 지정된 텍 파일이 데이터베이스에 삽입된다. 텍 파일의 이름이 템플릿 이름이 된다.

```
C:\>mytex -i foo.tex style_output=foo.sty image_output=foo.jpg
```

텍 파일에 포함된 다음과 같은 주석들이 템플릿 옵션으로 해석된다.

```
% description = An example of how to ...
% compiler = -c
% placeholders = 1
% defaults = foo
\documentclass{oblivoir}
...
```

다른 파일들을 함께 저장하려면 위 예와 같이 접미어 _output이 포함된 이름과 함께 파일을 지정하라. 그러면 데이터베이스에 다음과 같이 추가될 것이다.

```
[foo]
tex_output = foo.tex
tex = \documentclass{...}
style_output = foo.sty
style = \ProvidesPackage{...}
```

- u 추출된 텍 파일을 수정하고 그것으로 데이터베이스를 업데이트하려면 이 옵션을 사용하라.

```
C:\>mytex -u multilingual
```

- r 데이터베이스에서 지정한 템플릿이 제거된다.

-b 모든 템플릿들이 추출된다.

mytex.py가 이해하지 못하는 옵션들은 ltx.py에 전달된다

A.12 이따금 요긴한 스크립트들

scripts.db라는 데이터베이스 파일에 가끔 유용할 수 있는 스크립트들이 들어 있다. 대부분이 파이선으로 작성되었고, 일부는 파워셸로 작성되었다. myscript.py를 이용하여 특정 스크립트를 꺼내어 실행할 수 있다. 다음 예에서 scripts.db로부터 gencode.py가 추출되고, gencode.py가 KTUG 웹 주소를 가리키는 QR 코드를 만든다.

```
C:\>myscript gencode
C:\>gencode -q www.ktug.org
```

-I 새로 작성한 스크립트 파일을 scripts.db에 삽입한다.

```
C:\>myscript -I foo.py data_output=foo.tsv
```

다른 파일들을 함께 저장하려면 위 예와 같이 '_output'이 포함된 이름과 함께 파일을 지정하라. 그러면 데이터베이스에 다음과 같이 추가될 것이다.

```
[foo]
code_output = foo.py
code = import ...
data_output = foo.tsv
data = ...
```

-U 추출된 스크립트 파일을 수정하고 그것으로 scripts.db를 업데이트하려면 이 옵션을 사용하라.

```
C:\>myscript -U foo
```

-R 지정된 스크립트가 데이터베이스에서 제거된다.

-B 모든 스크립트들이 추출된다.

-C 실행 뒤에 추출된 스크립트가 삭제된다.

부록 B

응용

소프트웨어 제품에 대한 매뉴얼을 작성할 때 짐스러운 것이 UI 문자열이다. 대개 개발자들이 모든 UI 문자열을 담은 엑셀 파일을 제공한다. 그 파일은 매뉴얼 번역에 필수적이다. 예를 들어 “지금 설치” 버튼이 스페인어로 옮길 때 “Actualizar ahora”로 바뀌어야 한다.

	A	B	C
1	LID0000001	I am happy to join with you today in wha	우리나라 역사상 자유를 위한 가장 위대한 시위가 있었던
2	LID0000002	Five score years ago, a great American, in	백년 전, 오늘 우리가 서있는 자리의 상징적 그림자의 주인
3	LID0000003	This momentous decree came as a great l	그 중대한 법령은 억압적 불평등의 불길에 타들어가던 수
4	LID0000004	It came as a joyous daybreak to end the l	그 법령은 그들의 길었던 구속의 밤을 종식하는 기쁨의 사
5	LID0000005	But one hundred years later, the Negro st	그러나 백년이 지난 후에도, 흑인들은 여전히 자유롭지 못
6	LID0000006	One hundred years later, the Negro is still	백년이 지난 후에도, 흑인들은 미국사회의 한 구석에서 여
7	LID0000007	And so we've come here today to dramati	그래서 이 수치스런 상황을 알리고 바꾸고자 우리는 오늘
8	LID0000008	In a sense we've come to our nation's cap	어떤 의미로는 수표를 현금으로 바꾸기 위해서 우리는 우
9	LID0000009	When the architects of our republic wrote	우리나라를 건국한 사람들은 헌법과 독립선언문에 숭고한
10	LID0000010	This note was a promise that all men, yes,	그 약속어음은 모든 사람들에게, 예, 백인들처럼 흑인들에

실제 화면을 확인할 수 없는 상황에서도 UI 문자열 파일이 유용하다. 하지만 엑셀 파일이 너무 많은 언어들을 포함하고 있다면 이용하기에 불편하다. wordig.py와 \ReadTSV 명령을 이용하여 그 불편을 크게 줄일 수 있다.

1. wordig.py를 이용하여 엑셀 파일에서 특정 칼럼들을 추출하여 UI.tsv 파일로 저장한다.
2. \ReadTSV{UI.tsv}를 포함하는 UI.tex을 작성하여 PDF를 만든다. 57 페이지 **TSV 또는 CSV 파일 읽기**를 보라.
3. wordig.py를 이용하여 만들어진 PDF 파일에서 문자열을 검색한다.

```
C:\>wordig -t -a="1,3-5" -o=foo.tsv foo.xlsx
```

-a 옵션을 사용하여 추출할 칼럼들을 지정할 수 있다. 아무 것도 지정하지 않으면 모든 칼럼들이 추출된다.

```
\documentclass[9pt]{hzguide}
\LayoutSetup{}
\HeadingSetup{article, parskip=5pt}
\setmainfont{HANDotum.ttf}
\RenewDocumentCommand\texttsv{m}{\footnotesize #1}}
```

```

\NewDocumentCommand\colbox{m}{\parbox[t]{0.42\textwidth}{\raggedright\small #1}}
\TSVsetup{
  tabs={0pt, 0.12\textwidth, 0.56\textwidth},
  renderers={\texttsv, \colbox, \colbox},
  interline={\dotfill\linebreak},
  verbatim=false
}
\begin{document}
\ReadTSV{1}
\end{document}

```

이 레이텍 문서를 latex.db에 저장하면 언어별로 다수의 PDF 파일들을 보다 쉽게 만들 수 있다. 90 페이지 [레이텍 템플릿 모음](#)을 보라.

```
C:\>mytex -i UI.tex
```

그리고 다음과 같은 명령을 포함하는 xls2pdf.cmd를 만든다.

```

wordig.py -t -a %3 -o %~n2.tsv %1
mytex.py -f -o %~n2.tex -s %~n2.tsv UI

```

```

C:\>xls2pdf.cmd UI.xlsx UI_kor_eng "1-3"
C:\>xls2pdf.cmd UI.xlsx UI_eng_fre "1,3,4"
C:\>xls2pdf.cmd UI.xlsx UI_eng_ger "1,3,5"

```

참고

파워셸에서는 cmd 명령에 인자를 전달하기 위해 따옴표에 억음 부호를 추가해야 한다. ("1-3")

그러면 UI_kor_eng.pdf, UI_eng_fre.pdf, UI_eng_ger.pdf가 만들어질 것이다.

.....		
LID0000017	This is no time to engage in the luxury of cooling off or to take the tranquilizing drug of gradualism.	지금은 '냉정하자' 라는 사치스런 말이나 점진주의라는 안정제를 취할 때가 아닙니다.
.....		
LID0000018	Now is the time to make real the promises of democracy.	지금은 민주주의에 대한 약속을 지켜야 할 때입니다.
.....		
LID0000019	Now is the time to rise from the dark and desolate valley of segregation to the sunlit path of racial justice.	지금은 어둡고 황량한 차별의 계곡에서 양지 바른 인종적 정의의 길로 나와야 할 때입니다.
.....		

PDF 파일을 열지 않고 wordig.py를 이용하여 문자열을 검색할 수 있다.

```
C:\>wordig -a "I have a dream" UI.pdf
```



```
PowerShell
(base) PS C:\temp> wordig -a justice UI_eng_kor.pdf
UI_eng_kor.pdf
Page 1: withering injustice.
Page 1: justice is bankrupt.
Page 2: riches of freedom and the security of justice.
Page 2: path of racial justice.
Page 2: quicksands of racial injustice to the solid
Page 2: Now is the time to make justice a reality for
Page 2: bright day of justice emerges.
Page 2: which leads into the palace of justice.
Page 4: be satisfied until justice rolls down like
Page 5: heat of injustice, sweltering with the heat of
Page 5: oasis of freedom and justice.
(base) PS C:\temp>
```


찾아보기

A

A3, 13
A4, 13
A5, 13
A5V, 13
Acrobat Reader, 56
\action, 20, 55
\ActionSetup, 20
 afterskip, 20
 beforeskip, 20
 delimiter, 20
 font, 20
 inline, 20
address, 70
adhoc, 42
admonition, 53
\AdmonitionSetup, 54
after, 32
AfterFront, 70
afterlist, 43
aftersignal, 52
afterskip, 20, 26, 29, 33, 34, 36, 52, 55, 66
align, 26, 33, 36, 56
align-short, 33
alternative_path, 76
Anaconda, 75
\annotate, 47
 breakable, 47
 closing, 47
 color, 47
 font, 47

 index, 47
 opening, 47
 raise, 47
 space, 47
 star, 47
 superscript, 47

\annotate*, 47
\anota, 47
arbitrary, 14
aritcle, 17
\arraystretch, 37
article, 17
ASD-STE100, 53

B

B5, 13
\BackCover, 69
BackImage, 69
BackLogoImage, 69
base, 42, 44, 48
baseline, 32
\baselinestretch, 17
before, 32
BeforeBack, 70
beforelist, 43
beforesignal, 52
beforeskip, 20, 26, 29, 33, 34, 36, 52, 55, 65
\begin{callout}*, 34
\begin{IllustEnum}, 31
\begin{IllustImage}*, 31
\begin{IllustImage}^, 29
\begin{ImageTable}*, 38

`\begin{options}*`, 44
`\begin{SpecTable}*`, 38
`\begin{Table}*`, 37
`bgcolor`, 22, 48
`bibtex`, 77
`BlankBack`, 71
`BlankFront`, 70
`boffset`, 22
`\BookmarkIndexHead`, 48
`\BoundedBoxSetup`, 52
`braces`, 65
`breakable`, 45, 47
`breakable-adhoc`, 52
`breakable-default`, 52
`broad`, 30
C
`callout`, 33, 55
`calloutenum`, 34
`\CalloutSetup`, 34
 `afterskip`, 34
 `before skip`, 34
 `column`, 34
 `font`, 34
 `label`, 34
 `rule`, 34
 `space`, 34
 `tab`, 34
`caption`, 27, 30, 36
`caption-align`, 30
`caption-fit`, 27
`caption-ignore`, 27, 30
`\CaptionSetup`, 33
 `afterskip`, 33
 `align`, 33
 `align-short`, 33
 `before skip`, 33
 `delimiter`, 33
 `font`, 33
`caption-short`, 27
`caption-verb`, 27, 30
`caution`, 53, 55
`\CellImageOffset`, 38
`\cellimg`, 38
`cftsection`, 17
`chapteralign`, 18
`chaptercolor`, 18
`chaptercontents`, 18, 19
`\ChapterContentsDisable`, 19
`\ChapterContentsEnable`, 19
`chapterfont`, 18
`chaptermark`, 17, 22
`chapternamesize`, 18
`chapternumbersize`, 18
`chapterpage`, 17
`chapterstyle`, 17
`chaptertitlesize`, 18
`chapterwider`, 17
`chinese`, 9
`\cirnum`, 34, 48
 `base`, 48
 `bgcolor`, 48
 `fgcolor`, 48
 `font`, 48
 `fontfeature`, 48
 `raise`, 49
 `sep`, 48
`cjkchapter`, 18
`\ClearWatermark`, 21
`\CloneTerms`, 44
`closing`, 47
`code`, 65
`codebox`, 66
`codedesc`, 66, 67
`\codeinput`, 66
`\coderead`, 10, 66
`\coderead*`, 66
`coderesult`, 65
`\CodeSetup`, 65
 `afterskip`, 66
 `before skip`, 65
 `codebox`, 66
 `file`, 66
 `font`, 66

- fontsize, 66
- language, 66
- linespacing, 66
- offset, 66
- resultbox, 66
- vbar, 66
- width, 66
- codewrite, 66
- color, 21, 43, 45, 47, 57
- column, 14, 34
- comma-separated values, 57
- compiler, 78, 92
- content, 23
- \CoverSetup, 69
 - address, 70
 - AfterFront, 70
 - BackImage, 69
 - BackLogoImage, 69
 - BeforeBack, 70
 - BlankBack, 71
 - BlankFront, 70
 - DocumentType, 70
 - FontI, 70
 - FontII, 70
 - FontSizeI, 70
 - FontSizeII, 70
 - FontSizeIII, 70
 - FontSizeIV, 70
 - FrontImage, 69
 - FrontLogoImage, 69
 - hook, 71
 - manufacturer, 70
 - note, 70
 - ProductImage, 69
 - PubYear, 70
 - revision, 70
 - subtitle, 69
 - title, 69
 - TitleAlign, 70
 - topskip, 69
- CPython, 75
- csquotes, 10

- CSV, 57, 84
- csv, 58

D

- danger, 53
- darkgray, 56
- \DecolorHyperlinks, 56
- \DecolorHyperlinks*, 56
- Default notification, 44
- delimiter, 20, 33, 43
- description, 42
- detex.tsv, 86
- directory separator, 85
- docenv.conf, 76, 80
- docstring, 77
- \DocumentSetup, 9, 59
- \DocumentSetup*, 59
- DocumentType, 70
- DOTALL, 84
- draft, 63, 78

E

- \email, 56
- \EnlargePage, 56
- enum, 31
- enumerate, 34, 41, 42
- enumitem, 41, 42
- enummarker, 42
- epstopdf.exe, 80
- evenfootcenter, 16
- evenfootleft, 16
- evenfootright, 16
- evenheadcenter, 16
- evenheadleft, 16
- evenheadright, 16
- extraskip, 29, 52

F

- fake, 27, 30
- \fakeimage, 32
- \fakeimg, 32
- fc-cache, 81
- fc-list.exe, 80

- fgcolor, 22, 48
- figure, 26
- file, 66
- final, 63, 78
- final_compiler, 78
- \FinalizerOff, 63
- \FinalizerOn, 63
- \FinalizerSetup, 63
 - draft, 63
 - final, 63
- firstpath, 28
- fitline, 26
- fitwidth, 26
- float, 26, 36
- float-placement, 26, 36
- flush, 26
- font, 16, 20, 21, 33, 34, 36, 42, 45, 47, 48, 57, 66
- fontfeature, 48
- FontI, 70
- FontII, 70
- fontinfo.py, 80
- fonts_list.txt, 80
- fontsize, 21, 36, 66
- FontSizeI, 70
- FontSizeII, 70
- FontSizeIII, 70
- FontSizeIV, 70
- footcenter, 16
- footinner, 16
- footleft, 16
- footnotestyle, 36
- footouter, 16
- footright, 16
- footrule, 17
- frame, 26, 29, 55
- FrameBox, 43
- framecolor, 55
- \FrontCover, 69
- FrontImage, 69
- FrontLogoImage, 69

G

- gap, 26, 29
- GhostScript, 79
- graphicx, 61

H

- headcenter, 16
- heading, 58
- \HeadingSetup, 17, 19
 - article, 17
 - cftsection, 17
 - chapteralign, 18
 - chaptercolor, 18
 - chaptercontents, 18
 - chapterfont, 18
 - chapternamesize, 18
 - chapternumbersize, 18
 - chapterstyle, 17
 - chaptertitlesize, 18
 - chapterwider, 17
 - cjkchapter, 18
 - linespacing, 17
 - nochaptername, 18
 - paragraphstyle, 18
 - parindent, 19
 - parskip, 19
 - sectionalign, 18
 - sectionfont, 18
 - sectionsize, 18
 - sectionstyle, 18
 - subsectionsize, 18
 - subsubsectionsize, 18
- headinner, 16
- headleft, 16
- headouter, 16
- headright, 16
- headrule, 17
- height, 22
- hide, 57
- \HidePending, 57
- hook, 14, 71
- hzguide

- language, 9
- minted, 10
- Noto, 10
- packageset, 9
- pairquote, 10
- property, 9
- styleset, 9
- template, 10
- hzguide.xdy, 77
- \hznulldim, 26, 34, 36, 52, 55
- hztemplate.tex, 10, 73

I

- .idx, 77
- IEC/ISO 82079, 53
- IfDoc, 59
- \ifdoc, 59
- \ifLang, 10
- IfLanguage, 10
- \IfLuatex, 10
- \IfVartwo, 15
- IfVartwoEnlarge, 14
- \IfXetex, 10
- i.ini, 77
- IllustEnum, 31, 55, 56, 73
- IllustImage, 29, 31, 32, 55
- \illustimage, 29
- \illustimage*, 31
- \illustimage^, 29
- \IllustImageSetup, 35
 - afterskip, 29
 - beforeskip, 29
 - broad, 30
 - caption, 30
 - caption-align, 30
 - caption-ignore, 30
 - caption-verb, 30
 - enum, 31
 - extraskip, 29
 - fake, 30
 - frame, 29
 - gap, 29
 - imagealign, 30
 - label, 30
 - legend, 30
 - minwidth, 30
 - position, 30
 - scale, 30
 - showfilename, 30
 - star, 30
 - textstyle, 30
 - valign, 30
 - voffset, 30
- image, 21, 25, 27, 28, 32, 55
- \image*, 27, 32
- imagealign, 30
- ImageMagick, 79
- \ImageSetup, 35, 55
 - afterskip, 26
 - align, 26
 - beforeskip, 26
 - caption, 27
 - caption-fit, 27
 - caption-ignore, 27
 - caption-short, 27
 - caption-verb, 27
 - fake, 27
 - fitline, 26
 - fitwidth, 26
 - float, 26
 - float-placement, 26
 - flush, 26
 - frame, 26
 - gap, 26
 - label, 27
 - landscape, 27
 - legend, 27
 - middle, 27
 - scale, 26
 - showfilename, 27
 - star, 27
 - vbar, 27
- ImageTable, 38, 73
- \img, 32

- \img*, 32
- \imgitem[], 44
- indent, 19
- index, 43, 45, 47
- indexcategory, 43
- \IndexingEnable, 47, 65
- \IndexingEnable*, 48
- indexsame, 43, 45
- Inkscape, 79
- inline, 20
- \input, 9
- \inputminted, 10
- interaction=batchmode, 76
- interline, 58
- interval, 18, 19, 22
- inverse search, 81
- i.py, 77
- ISO/IEC Guide 37:2012, 69
- .ist, 77
- \item[{}], 65
- itemindent, 43
- itemize, 41
- itemsep, 43
- iu.py, 79
- J**
- jargon, 44, 46
- JB5, 14
- K**
- komkindex, 77
- korean, 9
- L**
- label, 27, 30, 34, 36
- labelsep, 43
- labeltype, 42, 43
- labelvoffset, 43
- labelwidth, 43
- landscape, 14, 27, 36
- language, 9, 54, 66
- latex.db, 90
- latex.json, 46
- latexmk.exe, 76
- \LayoutSetup, 13, 15
 - A3, 13
 - A4, 13
 - A5, 13
 - A5V, 13
 - B5, 13
 - column, 14
 - hook, 14
 - JB5, 14
 - landscape, 14
 - letter, 13
 - lrmargin, 14
 - paper, 13
 - paperheight, 14
 - paperwidth, 14
 - showlayout, 14
 - showtrims, 14
 - slide, 14
 - stockheight, 14
 - stockwidth, 14
 - ulmargin, 14
 - ulratio, 14
 - vartwomargin, 14
- leftmargin, 43
- legend, 27, 30, 36
- letter, 13
- \LineImageSetup, 35
 - after, 32
 - before, 32
 - raise, 32
 - scale, 32
 - showfilename, 32
- linespacing, 17, 66
- \listimg, 27
- \listimg*, 28, 32
- \ListOfFigures, 19
- \listofqueries, 57
- local.conf, 81
- localtexmf, 62
- lrmargin, 14
- ltx.py, 76

lualatex, 76
luaotfload-tool, 81
LuaTeX, 10
lyrics, 67

M

macro, 42, 65
\macro*, 65
macros, 65
macrosingle, 42
main, 78
\MakeAlbum, 35
\MakeAlbum*, 35
manufacturer, 70
markdelimiter, 17
marker, 42
markersep, 42
mate, 28, 29
\mateimage, 28
\MateSetup, 28
 firstpath, 28
 mate, 28
 secondpath, 29
memoir, 9
\Menu, 46
\menu, 46
menukeys, 46
\mi, 46
middle, 27
minted, 10, 66
minwidth, 30
multiline, 18, 42, 43
multilingual, 91
myscript.py, 93
mytex.py, 90

N

\NewBoundedBox, 51–55
 aftersignal, 52
 afterskip, 52
 beforesignal, 52
 beforeskip, 52
 breakable-adhoc, 52

breakable-default, 52
extraskip, 52
signal, 52
style, 52
symbol, 52
titlefont, 52
type, 52
\NewConditionals, 10, 59
\newlist, 42
\NewTerm, 45, 46
 breakable, 45
 color, 45
 font, 45
 index, 45
 indexsame, 45
 \mi, 46
 star, 45
 \term, 46
 \ui, 46
 vbar, 46
\NewTerms, 42, 44, 65
 afterlist, 43
 base, 42
 beforelist, 43
 color, 43
 delimiter, 43
 enumerate, 42
 enummarker, 42
 font, 42
 index, 43
 indexcategory, 43
 indexsame, 43
 itemindent, 43
 itemsep, 43
 labelsep, 43
 labeltype, 42
 labelvoffset, 43
 labelwidth, 43
 leftmargin, 43
 marker, 42
 markersep, 42
 offset, 42

- signs, 44
- star, 43
- style, 43
- terms, 44
- topsep, 43
- UI, 44
- vbar, 43
- nochaptername, 18
- noitemsep, 41
- note, 53, 70
- notice, 53
- Noto, 10

O

- \ObjectSetup, 26, 34, 36, 52, 55
 - afterskip, 55
 - align, 56
 - beforeskip, 55
 - framecolor, 55
- oddfootcenter, 16
- oddfootleft, 16
- oddfootright, 16
- oddheadcenter, 16
- oddheadleft, 16
- oddheadright, 16
- offset, 42, 66
- \onelineskip, 56
- opening, 47
- op.py, 83
- options, 44
- \optselected, 45
- \optunselected, 45
- otfinfo.exe, 80

P

- packageset, 9, 10
- \PageStyleSetup, 15
 - chaptermark, 17
 - chapterpage, 17
 - evenfootcenter, 16
 - evenfootleft, 16
 - evenfootright, 16
 - evenheadcenter, 16
 - evenheadleft, 16
 - evenheadright, 16
 - font, 16
 - footcenter, 16
 - footinner, 16
 - footleft, 16
 - footouter, 16
 - footright, 16
 - footrule, 17
 - headcenter, 16
 - headinner, 16
 - headleft, 16
 - headouter, 16
 - headright, 16
 - headrule, 17
 - markdelimiter, 17
 - oddfootcenter, 16
 - oddfootleft, 16
 - oddfootright, 16
 - oddheadcenter, 16
 - oddheadleft, 16
 - oddheadright, 16
 - sectionmark, 17
- pairquote, 10
- paper, 13
- paperheight, 14
- paperwidth, 14
- \paragraph, 20
- paragraphstyle, 18, 19
- parindent, 19
- parskip, 19
- PATH, 75, 76
- \path, 46
- PATHEXT, 75
- pdfcrop.exe, 80
- pdftops.exe, 80
- Pending, 56
- \pending, 56
- \PendingSetup, 57
- Personal, 28
- pixels per centimeter, 79
- placeholders, 92

- polyglossia, 9
- position, 30
- PowerShell, 35
- ppc, 79
- preamble, 9
- \printerchaptername, 17
- ProductImage, 69
- property, 9
- PubYear, 70
- py.exe, 75
- python launcher, 75
- pythontex, 77

Q

- \query, 57

R

- raise, 32, 47, 49
- \ReadTSV, 57, 95
 - csv, 58
 - heading, 58
 - interline, 58
 - renderers, 58
 - space, 58
 - tabs, 58
 - verbatim, 58
- reference, 53
- reference manual, 20
- renderers, 58
- \RenewBoundedBox, 54
- \RenewTColorBox, 53
- \ResetImageCounter, 35
- resultbox, 66
- revision, 70
- right, 30
- rotate, 21
- rule, 18, 34

S

- \sbfamily, 49
- scale, 21, 26, 27, 30, 32
- scripts.db, 93
- secondpath, 29

- sectionalign, 18
- sectionfont, 18
- sectionmark, 17
- \SectionNewpageOff, 20
- \SectionNewpageOn, 20
- \SectionNewpageOn*, 20
- sectionsiz, 18
- sectionstyle, 18
- Security Type, 28
- sep, 48
- \SetListStyle, 41
- ShadeBox, 52
- shell-escape, 76
- showfilename, 27, 30, 32, 35
- \ShowGraphicspath, 61
- \ShowImageCount, 35
- \ShowImageFilename, 35
- \ShowImageFilename*, 35
- showlayout, 14
- \ShowPageLayout, 14
- \ShowPending, 57
- showtrims, 14
- signal, 52
- signal word, 54
- signs, 44
- Silent mode, 44
- singleline, 42
- single-source publishing, 58
- \skiplines, 56
- slide, 14
- space, 34, 47, 58
- SpecTable, 37, 73
- star, 27, 30, 31, 37, 43, 45–47
- stockheight, 14
- stockwidth, 14
- stretch, 37
- style, 23, 43, 52
- styleset, 9
- \Sub, 46
- \SubsectionNewpageOff, 20
- \SubsectionNewpageOn, 20
- subsectionsiz, 18

- subsubsectionsize, 18
- subtitle, 69
- Sumatra PDF, 56, 81
- \Sup, 46
- superscript, 47
- symbol, 52
- synctex=1, 76
- syntax highlighting, 66
- T**
- \Tab, 46
- tab, 34, 46, 58
- Table, 35, 36, 55
- table, 36
- \TableOfContents, 19
- \tableofcontents, 19
- \TableOfContents*, 19
- \tableofcontents*, 19
- TableSetup, 36–38
 - afterskip, 36
 - align, 36
 - beforeskip, 36
 - caption, 36
 - float, 36
 - float-placement, 36
 - font, 36
 - fontsize, 36
 - footnotestyle, 36
 - label, 36
 - landscape, 36
 - legend, 36
 - star, 37
 - stretch, 37
 - vbar, 37
- tabs, 58
- tab-separated values, 57
- tabto, 46
- tabular, 36
- tabularray, 36
- tabularx, 36
- TC, 9
- tcolorbox, 51

- technical writer, 9
- template, 10, 73
 - \tplimage, 73
 - \tplimagetable, 73
 - \tpllist, 73
 - \tplpara, 73
 - \tplproblems, 73
 - \tplprocedure, 73
 - \tplspectables, 73
 - \tpltable, 73
 - \tpltext, 73
 - \tpltopics, 73
- \term, 46
- Term A, 73
- Term B, 74
- Term C, 74
- terms, 44
- \TermSetup, 45
- \TermsSetup, 42
- TEXEDIT, 81
- texindy, 47, 77
- TEXMFHOME, 81
- texmfhome, 62
- text, 21
- TEXTEDIT, 81
- \textsb, 45, 49
- textstyle, 30
- \textsubscript, 46, 47
- \textsuperscript, 46, 47
- \texttsv, 58
- thumb index, 21
- \ThumbIndexEnable, 21
- \ThumbIndexSetup, 22
 - bgcolor, 22
 - boffset, 22
 - content, 23
 - fgcolor, 22
 - height, 22
 - interval, 22
 - style, 23
 - toffset, 22
 - vertical, 23

- width, 22
- xoffset, 22
- tight, 17
- \tightlist, 41
- title, 69
- TitleAlign, 70
- titlefont, 52
- \titleref, 56
- tlconf.cmd, 81
- tlconf.py, 80
- tlmgr.exe, 81
- toffset, 22
- topic, 21
- topsep, 43
- topskip, 69
- \tplimage, 73
- \tplimagetable, 73
- \tpllist, 73
- \tplpara, 73
- \tplpara*, 73
- \tplproblems, 73
- \tplprocedure, 73
- \tplspectables, 73
- \tpltable, 73
- \tpltext, 73
- \tpltopics, 73
- trapper.py, 90
- troubleshooting, 19
- TSV, 57, 84
- type, 52

U

- UI, 44
- \ui, 46
- ulmargin, 14
- ulratio, 14
- unit.py, 86
- \URL, 56
- url, 46
- \URL*, 56
- user task, 20

V

- valign, 30
- vartwo, 14
- vartwomargin, 14
- vbar, 27, 37, 43, 46, 66
- veelo, 17
- \verb, 66
- verbatim, 27, 30, 58, 66, 67
- \verbatiminput, 10
- vertical, 23
- \vitem{}, 65
- voffset, 30
- Volumes, 44

W

- warning, 53
- \watermark, 21
- \WatermarkSetup, 21
 - color, 21
 - font, 21
 - fontsize, 21
 - image, 21
 - rotate, 21
 - scale, 21
 - text, 21
 - x, 21
 - y, 21
- width, 22, 66
- \winpath, 46
- wordig.py, 84
- wrapfigure, 29
- wrapper, 75, 76

X

- x, 21
- .xdy, 77
- xeCJK, 9
- xelatex, 76
- XeTeX, 10
- xindex, 47, 77
- xindex-hz-ko.lua, 77
- xindex-ko.lua, 77
- xoffset, 22

Y

y, 21

ㄱ

간체, 9

简体, 9

개인용, 28

개체, 55

경고, 53

경고문 표어, 54

고스트스크립트, 79

구문 강조, 66

그림판, 90

ㄴ

독스트링, 77

ㄷ

루아텍, 10

ㄹ

문제 해결하기, 19

ㅁ

반달 색인, 21

번체, 9

繁體, 9

변이단, 14

변환 명세 파일, 84

變二段, 14

보안 형태, 28

ㅂ

사륙배판, 14

사용자 업무, 20

수마트라 PDF, 56

신국판, 13

싱글 소스 퍼블리싱, 58

ㅇ

아나콘다, 75

아크로벳 리더, 56

워터마크, 21

위험, 53

이미지매직, 79

인버스 서치, 81

잉크스케이프, 79

ㅅ

전제부, 9

주의, 53

중괄호, 65

지텍, 10

ㅇ

참고, 53

참조 설명서, 20

ㅋ

캡션, 26, 33

캡처 및 스케치, 90

콜아웃, 33

ㅌ

테크니컬 라이터, 9

텍 라이브, 79

ㅍ

파워셀, 35, 63

版, 70

폴더 구분자, 85

ㅎ

함초롬 도움, 49

화제, 21

미니멀리즘의 목표는 사용자에게 요구되는 수고를 최소화하는 것이다.