도은이아빠

# HLaTeX으로 아름다운 한글 PDF 문서 작성하는 법

# 차 례

**제 1 절 들어가는 말** **3**

**제 2 절 기본적인 사항** **3**

2.1 PDF의 목적: 인쇄인가 화면보기인가 . . . . . 3

2.2 기본 전략 . . . . . 4

**제 3 절 글꼴** **4**

3.1 글꼴 사용 일반 . . . . . 4

3.1.1 PDF에서 이용할 수 있는 글꼴의 종류 . . . . . 4

3.1.2 한글 PDF 만들기의 몇 가지 방법 . . . . . 6

3.2 한글 PDF의 글꼴 사용 선택 . . . . . 7

3.2.1 한글 TEX 시스템들의 글꼴 사용 문제 . . . . . 7

3.2.2 UHC 폰트와 트루타입 폰트 . . . . . 8

3.3 트루타입 폰트 설정 . . . . . 9

3.4 영문 및 수학 글꼴과의 관계 . . . . . 12

**제 4 절 하이퍼링크와 한글 책갈피(bookmarks)** **12**

4.1 하이퍼링크 . . . . . 13

4.2 책갈피 . . . . . 13

**제 5 절 문서의 레이아웃** **14**

5.1 종이 크기와 여백 . . . . . 14

5.1.1 종이 크기의 선택 . . . . . 14

5.1.2 비규격 종이크기의 PDF 만들기 . . . . . 15

5.2 글자 크기와 장평 . . . . . 17

5.3 행간과 자간 . . . . . 19

5.3.1 자간과 단어간격 . . . . . 20

5.3.2 행간 . . . . . 21

5.4 장절명령의 설정 . . . . . 23
5.5 면주와 페이지스타일 . . . . . 24

**제 6 절 그림과 색상** **25**
6.1 그림 넣기 . . . . . 25
6.2 색상 . . . . . 26

**제 7 절 그밖의 몇 가지 문제** **26**
7.1 PSTricks 문제 . . . . . 26
7.2 기울인 글꼴 문제 . . . . . 27
7.3 밑줄 긋기 . . . . . 28
7.4 상호참조 및 자동조사 . . . . . 28
7.5 완성형 밖의 한글 쓰기 . . . . . 30

**제 8 절 마치는 말** **32**

**부록 A hangul-k 패키지에 관하여** **33**

**부록 B 이 문서의 Preamble** **34**

**찾아보기** **41**

# 제 1 절 들어가는 말

KTUG을 중심으로 한글 PDF 문서를 작성하는 방법이 발전하였다. 그러나 아직까지 쉽게 읽을 만한 안내문서가 없어서 KTUG 문서화 프로젝트 팀에서 이 문서를 준비하였다. 이 글은 KTUG에서 제공하는 hangul-k, hlatex-interword, hsectsty, hsetspace, myulem 패키지를 이용하여[1] 고품위의 한글 PDF 파일을 제작하는 방법에 대해서 간략하게 설명한다. 관련된 주제들이 너무나 방대하고 경우에 따라서는 매우 전문적이기 때문에, 최종 사용자의 입장에서 당장 적용해볼 수 있는 "방법"의 제시에 중점을 두었다.

LaTeX에서 PDF를 얻는 일반적인 방법에 대해서는 KTUG의 FAQ를 참고하라.[2]

이 글에서는 Lambda(Λ)나 PDFLaTeX을 사용하지 않고 LaTeX으로 얻은 DVI 파일을 DVIPDFM$x$로 PDF 변환하는 방법을 사용하겠다. 한글 문자가 중심인 한글 문서를 작성하는 데 있어서는 이 방법이 가장 효과적이고 사용자의 다양한 요구를 충족시켜 주는 것이기 때문이다.[3]

# 제 2 절 기본적인 사항

## 2.1 PDF의 목적: 인쇄인가 화면보기인가

PDF 문서는 온라인 문서의 한 표준이 되었다. 생각해볼 것은 사용자가 PDF로 최종결과를 얻으려 하는 목적이 무엇인가에 있다. 인쇄용으로 사용할 문서는 화면보기에서 좋은 결과를 얻는 것과는 다를 것이다. 최근 들어서 PDF를 출력용으로 사용하는 경우가 많아졌다고 한다. 따라서 PDF를 인쇄에 사

[1] 이 글은 여기 열거된 h-시리즈 패키지의 사용설명서를 겸한다.

[2] http://faq.ktug.or.kr/mywiki/

[3] 예컨대, DVIPDFM$x$를 이용하여야만 한글 책갈피(bookmarks)를 얻을 수 있다. 그리고 텍스트의 검색과 추출이 가능한 PDF 문서를 얻을 수 있게 할 뿐아니라, 문서에 암호를 걸 수도 있다. 이런 모든 기능은 현재로서는 오직 DVIPDFM$x$만이 제공하는 기능이다. 물론 한글이 문제가 아닌 경우는 그렇지 않다.

용할 PostScript 대신 쓰기 위해서 작성하는 경우도 분명히 있을 것이다. 이 글에서는 인쇄만을 목적으로 하는 PDF에 대해서는 논외로 하려 한다.

인쇄에 사용할 목적이라면 모르지만 온라인 화면보기용으로 PDF를 작성한다면 주의해야 할 것이 몇 가지 있다.

① 텍스트 검색이 되어야 한다. 온라인 문서에서 원하는 단어나 어구를 찾을 수 없다면 얼마나 답답하겠는가?

② 하이퍼링크가 작동하여야 한다. 온라인 문서에 하이퍼링크가 없다면 진정한 의미에서 "온라인" 문서라 하기 어려울 것이다.

③ 책갈피(bookmarks)와 같은 문서내 탐색장치가 있으면 편리하다.

## 2.2 기본 전략

이 글이 채택하고 있는 문서 작업의 기본틀은 다음과 같다.

**Compiler:** LaTeX으로 컴파일하여 DVIPDFM$x$로 PDF 변환한다. PDFLaTeX 등을 사용하지 않는다.

**Viewer:** gv, xpdf 등을 사용하는 경우는 문제삼지 않는다. Adobe Reader 또는 Acrobat Reader로 화면보기하는 경우만을 고려하겠다.

**Packages:** HLaTeX-0.991과 hangul-k 패키지를 사용한다.

# 제 3 절 글꼴

## 3.1 글꼴 사용 일반

### 3.1.1 PDF에서 이용할 수 있는 글꼴의 종류

일반 사용자들은 글꼴에 예민할 수밖에 없다. 최종적으로 만들어진 PDF는 다양한 글꼴을 이용할 수 있는데, 어떤 글꼴을 사용하였느냐에 따라 사용자

의 직관적인 만족도가 상당히 많이 달라지는 것을 볼 수 있었다. 한글 글꼴과 관련하여 다음과 같은 몇 가지 글꼴이 PDF에서 사용된다.[4]

**비트맵 글꼴** Type 3 글꼴이라고도 한다. 기본적으로 점(pixel)의 집합으로 문자를 표현하는 글꼴이다. 이 글꼴의 해상도(resolution, dpi)가 프린터의 해상도와 일치한다면 출력물의 품위가 크게 차이가 나지는 않지만 화면보기를 할 때는 특히 Acrobat Reader 5.x 버전 이하로 볼 때 글자윤곽이 찌글거려서 독자의 불만을 사는 글꼴이다. 다만 6.0 이후 버전은 비트맵 글꼴도 비교적 잘 표현해주는 것으로 생각된다. 이 글꼴은 사실상 글자라기보다 그림에 가깝기 때문에 "텍스트"로 다루어질 수 없고, 그 결과 검색이나 추출은 불가능하다. TEX에서는 원래 PK 글꼴이라 불리는 비트맵 글꼴을 TEX 자신의 METAFONT로부터 얻어내어서 인쇄와 화면 디스플레이에 사용했다.[5] 인쇄 결과는 매우 훌륭하지만 한글과 같이 수많은 폰트 파일이 필요한 경우 해상도에 따라 매번 PK 픽셀 글꼴 파일을 만드는 것은 좀 지루한 일이었고, PDF에 포함시킨 결과는 그다지 만족스럽지 못했다고 할 수 있다.

**PostScript 글꼴** `*.pfb` 또는 `*.pfa` 확장명을 가지는 Type 1 글꼴이 대표적이다. 윤곽선 글꼴이며 PDF에 가장 잘 맞는 글꼴 형식인 듯하고, 화면보기의 상태도 좋다.

**TrueType 글꼴** 한글은 PostScript 글꼴이 아주 예외적으로만 사용되고 있다. 예컨대 전문 출력소에서 사용하는 글꼴은 PostScript 글꼴이다. 그러나 일반 사용자가 PostScript 글꼴을 대하기란 아주 어려운 일이고 따라서 보다 저렴한 TrueType이 널리 쓰이게 되었다. 대부분의 한

[4] 글꼴 자체에 대한 전문적인 논의는 관련된 사이트나 책을 참고하라. 여기서는 PDF로 만들어진 파일을 화면 디스플레이할 때 사용자의 직관적인 관점에서 각 글꼴이 어떠한 차이를 보이는지만을 기술한다.

[5] TEX의 폰트 사용방법에 대해서는 이 글에서 자세히 논의할 수 없다. 관련된 다른 자료들을 참고할 것.

글 글꼴은 TrueType 형태로 쉽게 구할 수 있다. 윈도 운영체제의 기본글꼴들도 TrueType이다.

**OpenType 글꼴** TrueType과 PostScript 글꼴을 모두 포괄하는 차세대 글꼴이지만 한글 글꼴에 관한 한 아직은 그다지 많지 않다.

### 3.1.2 한글 PDF 만들기의 몇 가지 방법

한글 PDF가 만들어지는 일반적인 상황을 생각해보자.

**워드 프로세서를 사용하는 경우.** MS Word나 흔글과 같은 워드 프로세서에서 문서를 작성한 다음 Acrobat Distiller를 이용하거나 그와 유사한 PDF 인쇄 프로그램, 예컨대 ezPDF와 같은 프로그램으로 인쇄하여 PDF를 얻어낼 것이다. 윈도 운영체제에서 실행되는 워드 프로세서라면 글꼴도 TrueType이 사용되었을 것이므로 PDF 변환 프로그램을 어떻게 설정했느냐에 따라 달라지기는 하겠지만 TrueType을 내장(embed)하거나 하지 않은 PDF가 만들어질 것으로 생각된다. 이것이 뜻대로 되지 않으면 아마도 Type 3 비트맵 폰트가 들어갈 것이다. 그러나 이 결과에는 사용자가 그다지 만족하지 않을 것 같다.

**PostScript 인쇄 결과를 이용하는 경우.** 보다 일반적인 PDF 작성 방법은 PostScript 출력(`*.ps` 또는 `*.prn`)을 적당한 PostScript 해석기(예를 들면 GhostScript)로 PDF 변환하는 것이다. 이 때는 원래의 PostScript 출력에 어떤 글꼴이 어떻게 사용되었느냐가 문제인데, 매킨토시의 QuarkXpress의 경우를 예로 들어 보자. QuarkXpress에서는 작업시에 화면용 글꼴이라는 것을 주로 사용하는데 이것은 같은 이름의 PostScript 글꼴을 인쇄에 사용할 것을 전제로 draft용으로만 사용하는 비트맵 글꼴이다. 이 경우 화면상의 작업용 글꼴을 내장(embed)해서는 아니될 것이므로 일관되게 글꼴을 내장하지 않는 PostScript 파일을 만들어내고 있다.

**LaTeX에서.** LaTeX 사용자는 dvips(k)에 의하여 PostScript 출력 파일을 얻을 수 있다. 이것을 예컨대 GhostScript의 ps2pdf 스크립트를 이용하여 PDF로 변환할 수 있다. 그런데 dvips(k)는 (아직) TrueType이나 OpenType을 처리하는 방법을 제공하지 않는다. 따라서 TrueType, OpenType은 PK 비트맵 픽셀 폰트 파일로 변환되어(이 때 ttf2pk와 같은 유틸리티가 사용된다.) PostScript 파일로 포함될 것이다.

DVIPDFM$x$나 PDFLaTeX은 TrueType과 PostScript 폰트를 모두 잘 지원한다. 그러나 한글 TrueType 글꼴 사용에 있어서 PDFLaTeX은 몇 가지 한계를 가지고 있다. 이것은 PDFLaTeX 자체의 문제라기보다 PDFLaTeX을 위한 한글 TrueType 사용 기술이 충분히 개발되지 않았기 때문이다. 기울인 글꼴(oblique)을 사용할 수 없고 텍스트 검색이 되지 않는다.

## 3.2 한글 PDF의 글꼴 사용 선택

### 3.2.1 한글 TeX 시스템들의 글꼴 사용 문제

hLaTeXp는 비록 METAFONT source를 제공하지 않았지만 매킨토시의 글꼴이었던 신명조, 화명조 글꼴(PostScript)로부터 추출된 PK글꼴을 제공하였다. 그리고 흔TeX은 윈도 TrueType 글꼴을 이용하였다. 이 두 프로그램은 사실상 거의 동일한 TeX 엔진을 탑재하고 있었는데 최종 출력된 글꼴의 품위는 어느 정도 사용자에게 만족을 주는 수준이었다고 할 수 있다. 특히 흔TeX으로부터 Acrobat Distiller를 이용하여 만들어낸 PDF 파일은 트루타입을 내장할 수 있었으므로 워드 프로세서 흔글로부터 ezPDF를 이용하여 얻은 PDF와 큰 차이가 없었다.

HLaTeX은 0.99 이후로 UHC 글꼴을 채택하고 있다. 이것이 기본글꼴이고 PDF로 만들었을 때는 PostScript이므로 비트맵 글꼴에서 얻어진 불만족스러운 결과를 줄일 수 있었다. 그러나 UHC 글꼴은 아무래도 상업용 글꼴이 익숙한 워드 프로세서 사용자들에게는 조금 “촌스럽게(?)” 보이는 점도 없지 않았던 모양이다.

결과적으로, hLaTeXp는 글꼴의 디자인 품위는 나쁘지 않았지만 PK 비트맵 픽셀 폰트밖에 이용할 수 없었으므로 고품위 PDF를 만드는 데 적합하지 않았고, 혼TeX은 윈도 TrueType을 이용하였으므로 글꼴의 품위나 PDF 출력결과는 괜찮았지만 프로그램 자체가 치명적인 문제점을 지니고 있었고, HLaTeX은 PostScript 한글 글꼴을 제공한다는 점이 매력적이었으나 글꼴의 디자인에 대해 불만족한 사용자가 일부 있었다는 정도로 정리할 수 있겠다.

이 모든 문제를 이제는 HLaTeX 위에 한글 TrueType을 이용함으로써 극복할 수 있게 된 셈이다.

### 3.2.2 UHC 폰트와 트루타입 폰트

우리는 한글 PDF가 문제이므로, 결국 다음과 같은 두 가지 가운데 하나를 선택해야 한다.

① PostScript Type 1 글꼴인 UHC 글꼴을 사용하는 것. 이 때는 글꼴이 내장된다.

② 한글 TrueType 글꼴을 사용하여 글꼴이 내장되거나 되지 않은 PDF를 만드는 것.

첫번째 방법을 사용하게 되면 UHC 글꼴의 특성을 고려하지 않을 수 없다. 은광희 님의 UHC 글꼴은 현재 공개되어 있는(GPL 라이센스) 유일한 PostScript 한글 글꼴이다. 그런데 이 글꼴은 자소조합 글꼴이라는 특징을 가지고 있다. 화면상으로 글자들이 조금 비뚤비뚤하게 보이는 이유는 이러한 특성 때문이다. 그리고 PostScript 인쇄기가 Font Caching을 하는 경우에는 일부 캐시된 글자들이 뭉개지는 문제가 있는 것으로 보고되어 왔다. 해결방법이 없는 것은 아니지만 프린터의 설정을 일일이 맞추도록 요구하는 것은 쉬운 일이 아니었으므로, 그런 불편을 감수해야 한다. 또, 자소 조합 글꼴이기 때문에 이 글꼴을 내장한 PDF 파일에서는 검색과 추출이 불가능하

다. 이 때문에 UHC 글꼴로 만든 PDF 파일들에 대하여 인쇄상의 문제를 호소하는 경우가 상당히 많았다.

트루타입을 이용하는 것이 현재로서는 최선의 선택이 될 성싶다. 다만 트루타입은 LaTeX이나 HLaTeX 자체에서 기본으로 제공해주지 않는 것이므로 사용자가 자신에게 맞게 설정하여 사용해야 한다.

## 3.3 트루타입 폰트 설정

TrueType 글꼴로 PDF를 만들기로 작정하였다면 TrueType을 HLaTeX에서 사용할 수 있도록 설정해주어야 한다.

**미리 만들어진 폰트 패키지.** KTUG에서는 TrueType 폰트 사용 기술이 집중적으로 발달하였다.[6] 그 작업의 파생물로서, TEXMF TREE의 형태로 묶어져서 배포되는 TrueType 폰트 패키지들이 만들어졌다. 대표적인 것이 아시아폰트 기본팩이며, 윈도 기본글꼴 사용을 위한 패키지, 문화부 글꼴 패키지 등이 있다. 이 TrueType 글꼴 패키지들은 KTUG FAQ 사이트에서 찾을 수 있다.

압축파일 형태로 제공되는 글꼴 패키지들을 내려받아서 새로운 TEXMF TREE로 등록하고[7] `dvipdfmx.cfg`에 `map` 파일을 추가해주는 것으로 이 글꼴들을 사용할 준비는 대부분 끝난다. 한 가지 예만 들어두겠다. 이 과정은 자신이 사용하는 TeX 배포판에 따라 달라진다는 점을 기억하자.

아시아폰트 기본팩을 다운받아서 `C:\texmf-asiafonts-basic`에 풀어놓고 파일네임 데이터베이스를 갱신하였다면,

[6] 주로 조진환(ChoF) 님의 DVIPDFM$x$ 작업이 이러한 결과를 가져왔다. 여기에 기여하신 분들로는 박원규, 김도현, 신정식 님 등이 있다.

[7] 이 작업은 TeX 배포판에 따라 하는 방법이 다르다. MiKTeX의 경우에는 MiKTeX Options를 실행하여 `Roots` 탭에서 원하는 TEXMF TREE의 root를 찾아서 추가하고 `Refresh Now`를 실행하면 된다.

```
#> kpsewhich --format="other text files" \
       --progname="dvipdfm" dvipdfmx.cfg
```

이 명령으로 dvipdfmx.cfg의 위치를 확인하여 그 파일을 연 다음 마지막에

```
f asiafonts-basic.map
```

이 한 줄을 추가해준다.

특히, 박원규 님에 의하여 UHC 글꼴이 TrueType으로 포팅되었는데 이것을 은글꼴이라 한다. 은글꼴도 위와 같은 패키지 형식으로 배포된다. UHC 폰트를 좋아하는 분이라면 이 글꼴을 채택하는 것을 권장한다.

**사용자 자신의 폰트 설정** 김도현 님은 TrueType 설정을 보다 쉽게 하는 TTF2HLaTeXFont라는 perl script를 제작하였다. 이 스크립트를 사용하면 훨씬 자유롭게 여러 트루타입 글꼴을 자신의 문서에서 사용할 수 있다. 이 스크립트의 사용법은 아주 간단해서 몇 가지 폰트 결합을 config 파일에 지정하고 실행하는 것으로 끝이다. 심지어 TEXMF Tree의 형태로 설치해 주는 일까지 해준다. 사용방법은 스크립트의 배포처를 참고하라.

DVIPDFM$x$를 사용하면 한글 TrueType 글꼴을 내장(embed)하지 않도록 설정할 수 있다.[8] TEXMF TREE 묶음 형태로 배포되는 글꼴 패키지 가운데 mskttfonts라는 윈도 기본 글꼴 사용 패키지는 기본값이 글꼴을 내장하지 않도록 설정되어 있다. 이 방법의 장점은 만들어진 PDF의 크기가 정말 작다는 것이다. 그러나 사용자의 컴퓨터에 그 문서에 사용된 글꼴이 이미 설치되어 있어야 동일한 모양으로 디스플레이된다는 점이 한계이다. 한글 윈도 기본글꼴을 이용하는 위의 패키지로 작성된 문서는, 한글 윈도 기본글꼴

[8] 이 설정은 map 파일을 수정하면 된다. 구체적인 방법에 대해서는 다른 문서를 참고하라.

이 없는 환경에서는, 만약 Adobe Reader가 한국어 패키지를 설치하고 있는 상태라면 기본 글꼴로 대치해서 화면에 보여주기는 하겠지만, 동일한 상태로 보여진다고 장담할 수는 없다.

요컨대, 글꼴 내장하는 방법은 화면보기와 인쇄에서 언제나 동일한 형태를 유지할 수 있지만 파일크기가 커지고, 글꼴을 내장하지 않으면 파일의 크기는 작아지지만(그러나 많은 그림을 포함하고 있다면 큰 차이가 나지는 않을 수도 있다), 폰트를 가지고 있지 않은 사용자에게서 열리지 않을 가능성이 있다는 것이 문제이다.[9]

이로써 글꼴 결합이나 설정이 자유로와졌지만 자유에는 책임이 따르는 법이다. 만약 과도하게 많은 글꼴을 무분별하게 씀으로써 문서의 품위가 낮아지고 가독성을 떨어뜨리는 것은 전적으로 사용자 자신의 책임이다.

참고로, 이 문서는 이 분야에서는 이제 고전이 된 "HLATEX에서 임의의 한글 TrueType 글꼴 사용하기"라는 문서에서 제시된 글꼴 결합을 기초로 만들어졌다. 이 문서는 TTF2HLaTeXFont가 나오기 이전에 작성된 것이므로 굳이 따라하여 볼 필요는 없다.

마지막으로 글꼴의 설정과 사용에 대해서 추가할 것이 있다. 일반적인 교훈은 "하나의 문서에 세 종류 이상의 글꼴을 사용하지 말라"는 것이다. 그 세 종류란, 본문글꼴(\rm), 산세리프글꼴(\sf), 타입라이터(모노스페이스) 글꼴(\tt)를 가리킨다고 할 때, 적당한 한글 글꼴을 여기에 대응시켜서 사용하는 것이 논리적으로 일관성이 있는 것일 터이다. 필요하다면 강조(\emph) 명령에 쓰일 글꼴(영문에서는 이탤릭)을 별도로 설정할 수 있을 것이다.

[9] 최근의 Adobe Reader 6.0 이후 버전은 필요한 파일을 인터넷에서 다운받아서 설치하고 보여주는 기능을 내장하고 있다. 그러나 누구나 Adobe Reader를 쓰는 것도 아니며, 어디서나 6.0 이후 버전이 설치되어 있는 것도 아니고, 항상 인터넷에 연결되어 있지 않은 곳도 있는 것이다.

### 3.4 영문 및 수학 글꼴과의 관계

한글 문서를 작성할 때는 한글 글꼴의 선택도 중요하지만 영문 글꼴도 주의 깊게 선택하여야 한다. 그 이유는 HLaTeX이 한글 문서의 문장부호와 숫자에 영문 글꼴을 적용하고 있기 때문이다. 그 결과 어떤 영문 글꼴을 선택하였느냐에 따라 문서의 느낌이나 품위가 달라지게 된다.

수식을 많이 쓰는 문서를 작성한다면 수학 글꼴에도 고민을 좀 해야 할 것이다. 한글 본문 글꼴과 가장 잘 어울리는 영문 글꼴은 어떤 것이 있는지 경험을 통해서 직접 자신의 취향에 맞는 것을 찾아가는 도리밖에 없다.

가장 쉬운 선택은 "선택을 하지 않는 것"이다. 이 경우에는 Computer Modern 글꼴이 사용될 것인데, PDF 제작을 위해서는 되도록 Type 1 Scalable CM 폰트를 사용하는 것이 좋을 수도 있을 것이다. 이 경우 TypeWriter 서체는 아주 훌륭하고, 수학 글꼴과의 조화도 고민할 필요가 없다.[10]

영문 글꼴과 수학 글꼴에 대해서는 KTUG FAQ의 TeX 글꼴 페이지와 수학용 글꼴 페이지에 읽을 만한 정보가 있다.

이 문서에서 사용하고 있는 영문 글꼴은 **bera 글꼴**이다. 위의 사이트에서 관련된 정보를 얻을 수 있을 것이다.

## 제 4 절 하이퍼링크와 한글 책갈피(bookmarks)

하이퍼링크와 한글 책갈피는 PDF 문서를 풍부하게 만드는 요소이다. hyperref 패키지를 사용하여 구현한다. 주의할 것은 hyperref 패키지를 로드할 때 `[dvipdfm]` 옵션을 주라는 것이다. 안타깝게도 아직까지 `[dvipdfmx]`라는 옵션은 없지만 DVIPDFM$x$를 사용하는 것을 명시해두는 것이 좋다.

[10] 최근의 대부분의 배포판은 DVIPDFM$x$ 또는 PDFLaTeX을 위한 폰트 정의의 기본값으로 METAFONT CM이 아닌 Type 1 CM을 사용하도록 설정해두고 있다. 그러므로 PDF를 만드는 상황에서는 Type 1 CM에 특별히 신경쓸 이유가 없는 경우가 많다. 다만, dvips(k)를 이용하여 PS 파일을 먼저 만드는 경우에는 `-P pdf` 옵션을 지정하면 된다.

## 4.1 하이퍼링크

그 이후, 하이퍼링크는 패키지가 제공하는 기능을 써서 구현한다. 예를 들어 KTUG 사이트로 외부 하이퍼링크를 걸려면

```
\href{http://www.ktug.or.kr}{KTUG}
```

와 같이 하면, KTUG에서 보는 것처럼 외부 하이퍼링크가 만들어진다.

hyperref의 다양한 기능을 이용하면 하이퍼링크된 텍스트의 형태나 색상 등을 조절할 수도 있고 그밖에도 여러 가지 재미있는 기능을 구현할 수 있다. 더 자세한 것은 hyperref의 패키지 문서나 다른 LaTeX 관련 문서를 참고하라.

## 4.2 책갈피

책갈피(bookmarks)는 Adobe Reader 또는 xpdf 프로그램에서 지원하는 화면 디스플레이 기능의 하나이다. hyperref 패키지는 이 책갈피를 자동으로 만들어준다. 그렇지만 LaTeX으로 한글 책갈피를 넣는 일은 오랫동안 해결되지 않았던 문제였다. hyperref이 한글이 포함된 문자를 책갈피로 처리하는 데 어려움을 겪었기 때문이다.

한글 책갈피 만들기는 전적으로 DVIPDFM$x$의 기능이다. 따라서 예컨대 PDFLaTeX이나 GhostScript의 ps2pdf 등으로는 이런 결과를 (아직) 얻을 수 없다. 그리고, 한글 책갈피가 오류 없이 작동하려면 원칙적으로 한글 윈도 운영체제 상에서 한글 Adobe Reader를 사용하여야 한다. 다른 상황에서 되는 경우도 있겠지만, 여기서는 문제삼지 않겠다.

한글 책갈피 설정을 위해서는 Preamble에 다음과 같이 써넣고 컴파일하는 것으로 충분하다. 나머지는 DVIPDFM$x$가 알아서 해준다.

```
\usepackage[dvipdfm,CJKbookmarks]{hyperref}
\AtBeginDvi{\special{pdf: tounicode KSCms-UHC-UCS2}}
```

책갈피를 만들지 않으려면 다음과 같이 하라.

```
\usepackage[dvipdfm,bookmarks=false]{hyperref}
```

이 책갈피 만들지 않기는 DVIPDFM$x$가 나오기 전 한글 책갈피 때문에 어려움을 겪던 시절에 권장되던 방법이다. 이렇게 하지 않으면 컴파일 과정에서 무수한 경고가 쏟아져나오고 한글이 "깨진" 책갈피가 만들어지곤 했었다.

이 당시의 또다른 해결책으로 bmsec 패키지를 이용하여 한글 문서에서 영문 책갈피를 만드는 방법도 있었는데, 이것은 \chapter, \section와 같은 장절명령에서 특별한 옵션 인자로 영문 책갈피 문자열을 따로 지정해주는 방식이었다. 이 방법은 지금도 필요한 경우가 없지 않을 것이다.

# 제 5 절 문서의 레이아웃

## 5.1 종이 크기와 여백

### 5.1.1 종이 크기의 선택

종이 규격 페이지에 보면 인쇄에 쓰이는 종이 규격에 대한 설명이 있다.

온라인 문서라 하더라도 인쇄를 의식하지 않을 수 없으므로 익숙한 규격 용지에 맞추어서 작성하는 것이 나쁘지 않은 선택이다. 예컨대 A4 사이즈나 B5 사이즈는 전형적인 종이 규격에 해당할 것이다. 그런데 때로는 규격 외 사이즈로 문서를 만들어야 할 때가 있다.

LaTeX의 레이아웃 기본값은 전체적으로 보아서 상당한 기준의 요구를 충족시킨다. 그러나 여백은 많은 사람들이 너무 과도하게 주고 있는 것이 아니냐는 느낌을 받는 듯하다. 예를 들어 [a4paper,10pt]의 값으로 문서를 조판하면 \textwidth는 345포인트 정도가 된다. 이 값은 약 45mm 정도의 hmargin 값을 준 것과 비슷하다. 이 정도의 넉넉한 여백이 있어야 \textwidth가 지나치게 길어지지 않고 가독성을 높일 수 있다. 여백을 줄이고자 할 때는 여백을 줄인 결과가 독자의 독서를 방해하지는 않을 것인지 한 번 더 생각해보기 바란다.

종이 크기와 텍스트 영역은 서로 밀접한 관계가 있다. 이것은 geometry 패키지를 이용하여 쉽게 구현할 수 있다. 예를 들면 이 문서는 그림 1과 같은 방법으로 종이 크기와 텍스트 영역을 설정하였다. layouts 패키지를 이용하면 현재의 디자인 상태를 좀더 잘 확인할 수 있다. layouts를 이용하여 이 문서를 점검한 결과에 대해서는 그림 2를 참고할 수 있다.

```
%% PaperSize Setting.
\usepackage[paperwidth=150mm,
    paperheight=212mm,
    hmargin=20mm,
    top=25mm,
    bottom=27mm]{geometry}
\setlength\headheight{27.5pt}
```

그림 1: 이 문서의 레이아웃 설정

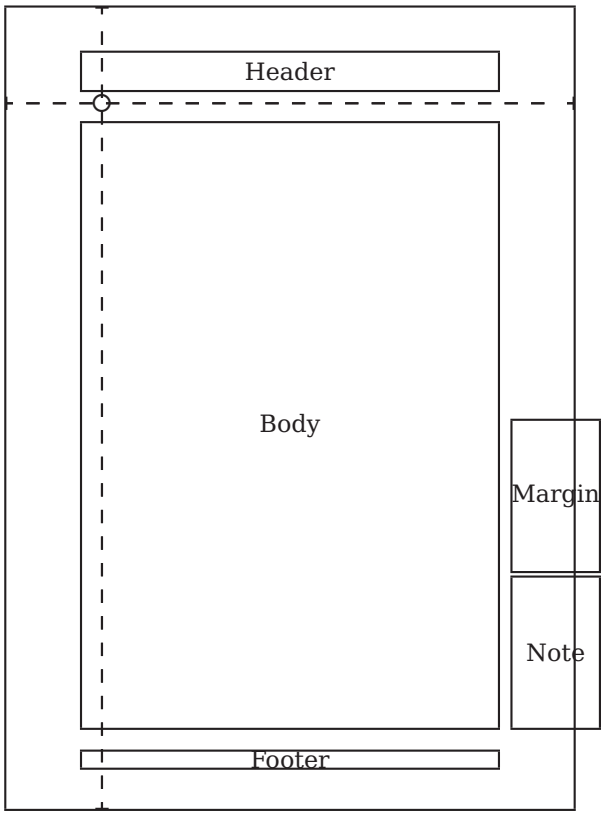
### 5.1.2 비규격 종이크기의 PDF 만들기

온라인 문서를 작성하면서 선택할 수 있는 좋은 방법은 A4 사이즈보다 종이 자체의 크기를 작게 하는 것이다. 그러면 여백을 줄이면서도 판면의 가로 길이(\textwidth)를 지나치게 늘리지 않을 수 있을 것이다.

이런 식으로 새로운 크기를 정했을 때, PDF 문서의 크기를 정해주려면 몇 가지 신경쓸 점이 있다.[11]

먼저, 인쇄되는 종이는 규격 용지로 하고 사용자가 설정한 (그보다 작은) 판면 설정을 다만 표시만 해주는 방법이 있다. 예를 들면 A4 사이즈의 종이에 이 문서를 찍으면 어디까지가 설정된 페이지의 경계가 되는지를 frame으로 그려준다든가 하는 방법이다. 이 목적을 위해서는 crop 패키지가 쓰인다.

[11] PDFLaTeX은 geometry 패키지를 읽어서 geometry가 설정한 종이와 판면 크기를 그대로 PDF로 만들어주기 때문에 매우 편리한데 DVIPDFM$x$는 별도의 절차가 필요하다.




Lengths are to the nearest pt.

| | |
|---|---|
| page height = 603pt | page width = 427pt |
| \hoffset = 0pt | \voffset = 0pt |
| \oddsidemargin = -15pt | \topmargin = -38pt |
| \headheight = 28pt | \headsep = 25pt |
| \textheight = 455pt | \textwidth = 313pt |
| \footskip = 30pt | \marginparsep = 11pt |
| \marginparpush = 5pt | \columnsep = 10pt |
| \columnseprule = 0.0pt | |

그림 2: layouts 패키지로 그려본 이 페이지의 디자인

또다른 방법은 PDF 자체의 크기를 설정된 페이지에 맞추는 방법이다. 결과는 geometry로 설정한 paperwidth, paperheight 값에 따라 PDF 문서가 만들어지게 될 것이다.

만약 DVIPDFM$x$ 최신 버전(20040320 이후 버전)을 사용한다면, 이 문제는 보다 쉽게 해결된다.[12] -p 옵션으로 종이 크기를 지정해주는 것이다. DVIPDFM$x$는 수많은 기본 종이값을 알고 있으므로 그 가운데서 선택하여도 좋고, 아니면 직접 가로x세로 길이를 지정해도 된다.

```
#> dvipdfmx -p "150mm,212mm" foo
```

그러나 최신 버전이 아닌 이전 버전의 DVIPDFM$x$를 쓰고 있다면 이 때는 dvipdfm에 적용하던 방법을 이용한다. 문서의 첫 부분에 다음과 같은 명령을 포함한다.

```
\AtBeginDvi{%
  \special{pdf: pagesize width 150mm height 212mm}}
```

그리고, 반드시 geometry **설정을** hyperref **설정보다 앞에 두도록** 하고, hyperref 설정 이후에 다음 그림 3의 코드를 포함한다.
이렇게 하면, 명령행에서 dvipdfmx 만을 실행하여도 적당한 크기의 PDF가 만들어지도록 할 수 있다.

## 5.2 글자 크기와 장평

영문의 경우와 달리 한글 글자는 모두 네모꼴이라서 글자의 가로폭이 비교적 균일하다는 특징이 있다. 영문의 경우는 variable width(가변폭) 글꼴이 대부분 본문 글꼴로 쓰이고 있기 때문에 커닝(kerning)이라든가 스페이싱이 매우 정교하게 발달하였지만, 한글 문서의 경우는 이 분야에서 그다지 많은 연구가 이루어지지 않았다.

[12] 이 기능이 추가된 최신 버전의 MiKTEX 바이너리 실행파일은 아직까지 구할 수 없다. 앞으로 나오겠지만 현재로서는 W32TEX에서만 윈도용 바이너리를 볼 수 있다. 유닉스나 리눅스 또는 CygWin 사용자라면 소스를 내려받아서 컴파일-빌드하여 사용할 수 있을 것이다.

```
\makeatletter
\AfterBeginDocument{%
  \ifx\special@paper\@empty\else
    \ifHy@setpagesize
      \special{papersize=\special@paper}%
    \fi
    \Hy@DisableOption{setpagesize}%
  \fi
}
\makeatother
```

그림 3: 임의의 종이 크기를 설정하기 위한 코드

활판 인쇄된 책들을 보면 글자는 네모꼴의 디자인 영역 안에 비교적 작게 들어앉고 글자와 글자 사이에는 조금 넉넉한 여유분이 있다. 그러나 전자조판된 책들은 글자와 글자 사이가 거의 여유가 없을 정도로 붙고 그 대신 단어와 단어 간격은 이전보다 더 벌어져 있는 것을 육안으로 확인할 수 있을 것이다. 이 "마이너스 자간"의 관행은 전자출판에서는 거의 확립된 조판 방식이 되었지만 그 효과가 과연 바람직한가에 대해 신뢰할 수 있는 통계적 증거가 아직은 없다고 생각된다. 즉, 글자와 글자를 붙인다고 해서 가독성이 높아지는 것도 아니라는 것이다. 단어별로 책을 읽게 되는 것이 아니라 일정한 어구를 한꺼번에 파악하면서 읽는 것이 일반적인 우리나라에서 문자폭의 절반 가까운 단어간 간격을 주는 것이 꼭 바람직한지는 잘 알 수 없다. 예를 들어 띄어쓰기에 있어서도 의존명사 앞에 오는 스페이스를 어절이 끝난 다음의 스페이스보다 더 적게 주는 것이 가독성을 높여주는 것은 아닐까? 아무튼 이런 점에 대해서 좀더 연구가 이루어졌으면 좋겠다.

TTF2HLaTeXFont 스크립트로 `*.ttf`로부터 `*.tfm`을 추출하면 c 시리즈 글꼴을 정의해준다. LaTeX NFSS의 c 시리즈 글꼴은 장평 75%의 글자체를 의미하는데, HLaTeX의 UHC 글꼴에도 이 시리즈가 정의되어 있다. 그러나 TTF2HLaTeXFont의 c 시리즈 글꼴은 장평 92%의 글자체를 의미하는 것으

로 바뀌어 있다.[13] 한글에 대해서만 이 시리즈의 글꼴을 사용하도록 하려면 \makethinhangul이라는 명령을 선언해주면 된다.[14] 이 명령이 하는 기능은 \hfontseries{c}를 선언해주는 것뿐이다.

내부적으로 글꼴을 새롭게 설정하는 명령들이 있다. 예를 들면 각주나 그림과 표의 캡션, 장절명령, 색인(찾아보기)을 만드는 theindex 환경, 표지를 만드는 titlepage 환경이 대표적이다. 이러한 환경 또는 명령 안에서는 \makethinhangul이 작동하지 않으므로, 만약 이러한 곳에서도 c 시리즈 글꼴이 사용되기를 원한다면 거기에 맞추어서 \makethinhangul을 선언해주면 된다. 이 문서에서는 예컨대 각주에서도 c 시리즈 글꼴을 사용하기 위해 다음과 같이 선언하였다.

```
%% footnote thinhangul setting.
\let\origfootnotesize\footnotesize\relax
\def\footnotesize{\origfootnotesize\hfontseries{c}}
```

c 시리즈 글꼴을 채택할 것이냐의 여부는 전적으로 개인적인 취향의 문제이다. 필자가 선호하는 것은 c 시리즈 글꼴을 사용하면서 본문을 11포인트로 하는 것이다. 10포인트에 92% 장평을 적용하면 글자가 조금 작아보이는 단점이 있는데, 화면용 PDF 파일의 작성에는 이 정도(11포인트 본문 + 92% 장평)의 설정이 가장 보기 좋다고 생각한다. 이밖에, 좀더 워드 프로세서에 가까운 느낌을 주려면 본문을 10.5포인트로 하는 방법도 있는데, 이것은 size10d5.sty라는 스타일을 사용해서 구현할 수도 있다.

## 5.3 행간과 자간

애초 HLaTeX은 행간이나 자간에 대한 고려가 거의 없었다. 영문 문서의 설정을 그대로 사용하게 되어 있었던 것이다. 그러나 타이포그라피의 측면에

[13] KTUG의 정기 모임에서 이런 방식의 c 시리즈 글꼴을 채택하자는 비공식 논의가 이루어진 바 있었고, 최초로 시도된 것은 아시아폰트 기본꾁에서였다.

[14] \makethinhangul 명령은 LaTeX이나 HLaTeX의 기본명령이 아니고 TTF2HLaTeXFont 스크립트에 의하여 생성되는 xxttf.sty에 정의되어 있다.

서 한글 문서의 행간과 자간은 영문 문서의 그것과 동일하면 문서가 지나치게 촘촘해 보여서 품위가 떨어진다.

hLaTeXp의 hangul 패키지는 기본 행간을 1.3으로 하는 한글 문서 설정 기본값을 제시했었다. hLaTeXp로 만들어진 문서가 그래도 읽기에 수월했던 이유가 아마도 이러한 기본 행간의 역할이 아니었을까 한다.

물론 행간은 사용자가 마음만 먹으면 얼마든지 쉽게 조절할 수 있다. 그러나 스타일의 기본값으로 제시되는 것이 있으면 아무래도 안심하고 쓸 수 있을 것이다. 사용자를 보다 편안한 문서 작성 환경으로 유도하는 효과도 있으리라고 생각한다.

### 5.3.1 자간과 단어간격

한글 문서에서 자간과 단어간격을 조절할 수 있게 하는 스타일 패키지는 hlatex-interword이다.

이 스타일은 `*.dtx`와 `*.ins` 형식으로 제공되므로 패키지 안내 문서를 `hlatex-interword.dtx`를 컴파일함으로써 얻을 수 있다. 간단히 필요한 것만 요약하면 다음과 같다.

**[default] 옵션.** 영문 문서의 경우와 동일한 자간을 사용하되 단어 간격을 조금 벌려주는 정도의 효과를 가져온다. 영문 문서 설정을 가져다 쓰는 경우보다 읽기에 조금 편안한 문서를 만들 수 있다. 이 문서가 이 옵션으로 작성되었다.

**[HWP] 옵션.** 흔글에서 작업할 때와 거의 유사한 단어 간격과 행나눔을 보여주는 옵션이다. 따라서 단어 간격은 상당히 넓은 편이다. 이 옵션을 더 좋아하는 분도 있으리라 생각한다.

**[narrower] 옵션.** 자간을 조금 촘촘하게 한다. "마이너스 자간"에 익숙한 분들이 좋아할 것으로 생각되고, 은글꼴을 본문 글꼴로 사용할 때는 꽤 좋

은 효과를 가져온다. 한양 계열의 본문 글꼴에서는 어울리지 않는다는 것이 개인적인 생각이다. [default] 옵션과 함께 쓸 수 있다.

**\interhchar 명령.** 자간을 임의로 설정하게 해준다. 예컨대 다음 명령은 자간을 -1포인트 좁혀준다. 숫자는 0.1포인트 단위이다.

```
\interhchar{-10}
```

**\interhword 명령.** 단어간격 설정이다. 이 명령을 사용할 때 주의할 점은 남용하면 특히 문서 중간중간에 영어 문장이 들어가는 경우 너무 단어 간격이 넓어 보일 수 있다는 점이다. 자간은 영어 문장이나 문자에 영향을 끼치지 않지만 단어간격은 그렇지 않다.

세 개의 길이(dimension)를 인자로 취하는데, 마지막의 두 인자는 허용치의 상하한을 표시한다. 예를 들어

```
\interhword{.3em}{.2em}{.3em}
```

이 명령은 단어 간격을 0.5em을 기준으로 최대 0.7 (0.5+0.2)em, 최소 0.2 (0.5−0.3)em으로 조판하라는 지시가 된다.

아주 특별한 경우가 아니면 위의 임의 설정 명령들은 쓰지 않는 쪽이 좋다. 되도록이면 기본값으로 사용하도록 하자.

### 5.3.2 행간

행간은 보통 \baselinestretch 승수를 바꿈으로써 double spacing을 얻는 것이 일반적이었다. 영문의 경우 double spacing은 \baselinestretch 값을 1.6으로 하는 것이 권장되고 있다.

```
\renewcommand\baselinestretch{1.6}
```

또다른 방법으로 setspace 패키지가 쓰인다. 한글 문서 작성 상황에서도 이 패키지를 쓰는 것이 불가능할 것은 없지만, 이 패키지는 그 성질상 floating objects(떠다니는 개체)와 각주 안에서 모든 행간격을 1.0으로 되돌리는 기능이 있는데, 한글 문서의 경우 각주의 행간격이 1.0으로 강제되는 것은 무리라 하여 이것을 조금 유연하게 다룰 수 있도록 하고 한글 문서의 표준 행간격을 제시하기 위해 작성된 것이 hsetspace 패키지이다.

hsetspace에서는 몇 가지 추가적인 옵션과 명령을 제공한다.

**[hangul] 옵션.** 이 옵션을 주면 문서의 기본 행간격을 1.333으로 맞추고 각주와 floats, 그리고 quote, quotation 환경 내의 행간격을 1.2로 조절해준다. 만약 각주 사이의 간격을 조절하겠다면 [adjustfootnotesep]이라는 긴 이름의 옵션을 [hangul] 이후에 추가할 수 있다.

또, 여기에 [adjustverbatim] 옵션을 추가하면, verbatim 환경 내에서도 좁은 행간격(기본값은 1.2)을 적용해준다. 다만 이 모든 기능은 반드시 [hangul] 옵션을 준 상태여야 한다는 것이다.

**[nofloatspacing] 옵션.** floats와 각주에 적용되는 setspace의 기본 기능(행간격을 1.0으로 맞추는 기능)을 끈다. 그러므로 이 옵션을 준 이후에는 문서 내의 모든 행간격이 문서의 기본 설정 행간격을 따라간다. 예컨대 [hangul] 옵션을 선언하였다면, 각주 내에서도 행간격은 본문과 동일하게 설정된다.

그러므로, 이 옵션을 추가한 경우에는 [adjust...] 옵션은 무의미하다.

**\SetHangulspace 명령.** [hangul] 옵션을 준 경우에 사용할 수 있는 명령으로서 한글 문서에서의 행간격을 일괄 지정할 수 있게 한다. 두 개의 인자를 가지며, 첫번째 것은 문서 전체의 본문 기본 행간격이 되고 두번째 것은 floats와 각주 내의 행간격으로 쓰이게 된다.

```
\SetHangulspace{1.2}{1.0}
```

위의 명령은 본문의 행간을 1.2로 하고, floats, 각주, quote, quotation 내의 행간격을 1.0으로 설정하도록 한다. 여기서 숫자는 \setstretch 값, 즉 \baselinestretch 값이다.

각주 사이의 간격은 이것으로 조절되지 않으므로 \footnotesep 등은 사용자가 직접 지시하여야 한다.

이 이외의 다른 기능은 setspace의 경우와 같다. 즉, spacing 환경을 써서 문서 일부의 행간격을 임의로 조절할 수 있다. 그러므로 hsetspace와 setspace를 동시에 사용하지 않도록 하라. 둘 가운데 하나만 있으면 된다. setspace 패키지의 사용법에 대해서는 패키지 문서를 참고할 것.

## 5.4 장절명령의 설정

장절명령은 HLaTeX의 어려운 점 가운데 하나이다. HLaTeX의 hangul 패키지는 장(chapter)뿐 아니라 절(section)에도 "제 1 절"과 같은 형식으로 "제"와 "절"을 붙였는데, 이 때문에 적어도 장과 절에 대해서는 다른 영문 문서에서 적용되는 많은 장절형식 관련 패키지가 부적절하게 작동한다. 그리고 간단히 \@startsection을 이용하는 영문 스타일의 절(section) 정의와도 잘 맞지 않아서 장절 형식을 바꾸고자 하는 사람들에게 많은 고통을 안겨주었다.

이 문제를 부분적으로 해결하고자 시도한 것이 hsectsty 스타일이다. 이 스타일은 sectsty을 한글화한 것인데 titlesec에 비해서 유연성이 조금 떨어지지만 그래도 장절의 폰트나 형식을 좀더 쉽게 바꿀 수 있다는 점 때문에 채용되었다. 장절명령에 대한 사항은 KTUG FAQ의 장절명령 페이지를 참고하기 바란다.

이 스타일은 기본적인 사용법이 sectsty과 같다. 그러므로 사용법 자체는 sectsty 패키지 문서를 참조하기 바란다. hsectsty를 로드하면 sectsty를 불러온다.

이 패키지를 사용하였을 때 기본값은 “제”와 “장” 또는 “절” 사이의 간격을 ~로 연결한 것보다 조금 좁게 잡아주는데, 이것이 마음에 들지 않아서 hangul의 원래 형식으로 되돌리려면 [THE] 옵션을 지정하면 된다.

추가된 옵션으로 [ensec]이 있다. 이 옵션을 주면 절(section)을 영문 문서의 절과 동일하게 식자해준다. 즉, “제”와 “절”이 붙지 않는다.

이 문서에서 hsectsty 패키지가 쓰였다. 관련된 코드를 보면 그림 4와 같다.

```
\usepackage{hsectsty} % 한글 장절명령 스타일.
\sectionfont{%
  \noindent\hfontseries{bc}\textcolor{blue}}
\subsectionfont{%
  \noindent\hfontseries{bc}\textcolor{DarkMagenta}}
\subsubsectionfont{%
  \noindent\hfontseries{bc}\sffamily\selectfont
  \textcolor{TempColor}}
```

그림 4: 장절명령 스타일 정의

여기서 쓰인 색상(blue, DarkMagenta, TempColor)에 대해서는 아래 색상 관련 제 6 절에서 더 자세히 알아본다.

## 5.5 면주와 페이지스타일

면주(面注)는 흔히 PageStyle에서 “running heading”이라고 불리는 것이다. 이것을 정의하는 데는 fancyhdr 패키지가 쓰인다. 이 패키지만으로 원하는 거의 모든 설정을 할 수 있기 때문에 여기서 면주에 대하여 더 자세한 사항은 생략하려 한다. FAQ FancyHdr 페이지를 참고하라.

한 가지만 지적하자면, LaTeX은 “고아와 과부(orphans and widows)”라는 재미있는 이름으로 불리는 외따로 떨어진 줄을 허용하지 않는 **훌륭한** 기능이 있어서 예컨대 한 줄만이 외따로 한 페이지의 마지막이나 처음을 만들

게 되면 그것을 다음 쪽으로 넘기는 경향이 있다. 이것은 조판상 올바르고 독자의 시선을 불필요하게 움직이지 않고 내용의 연결을 보장해준다는 점에서 반드시 적용되어야 할 문서 작성 규칙이지만, 워드 프로세서에 익숙한 일부 독자들은 앞 페이지의 아랫부분이 남거나 문단 사이가 조금 벌어지는 것을 한 줄, 심지어 절 제목 한 줄이 그 페이지의 마지막에 홀로 떨어지는 것보다 더 못참는 경향이 있는 듯하다.

이럴 경우 일부러 LaTeX의 규칙을 어기면서 "사회적으로 보호받아야 할 필요가 있는" 성분들을 다수 생산하는 것은 현명하지 못한 처사라 생각된다.[15]

다만, 앞 페이지를 종으로 채우고(vertically fill) 다음 페이지로 넘어갈 때 문단 간의 간격이 벌어지는 것을 못 참겠는 분을 위해서 `\raggedbottom` 명령 하나를 지적해두고 가겠다. 이 명령을 지정하면 페이지의 아래쪽 끝을 일부러 채우지 않는다. 그러므로 일부 문단이 다음 쪽으로 넘어가면 문단 사이의 간격을 벌리지 않고 페이지 아래를 비운다.

# 제 6 절 그림과 색상

## 6.1 그림 넣기

LaTeX에서의 그림 처리에 대해서는 훌륭한 문서들이 많다. KTUG의 GFaq도 그 중의 하나이다.

DVIPDFM$x$는 GhostScript를 이용하여 EPS 그림파일을 PDF 그림으로 변환하여 처리한다. 따라서 GhostScript가 잘 설치되어 있고 그림에 문제가 없다면 EPS 그림파일을 이용할 수 있다.

PDF나 JPG 또는 PNG 그림들을 그대로 이용할 수도 있다. 이 형식의 그림들이라면 ebb를 미리 실행하여 `*.bb`라는 확장명을 갖는 Bounding Box 파

[15] 그래도 이 기능을 반드시 꺼야겠다는 분을 위해서, 이것은 TeX의 설정값 가운데 하나인 `\clubpenalty`와 `\widowpenalty` 값을 변경하여 동작을 제어할 수 있다는 점만 지적해두고자 한다. LaTeX은 이 값을 각각 150으로 정해두고 있다.

일을 미리 얻어두어야 한다는 점만 조심하면 될 것이다. 더 자세한 것은 여러 그림 처리 관련 문서를 참고하거나 graphicx 패키지 문서를 보라. 그림 처리는 기능 그 자체의 문제라기 보다는 만들어진 그림 자체의 품질이나 그림 처리의 "아이디어"가 더 큰 문제가 될 때가 많다.

## 6.2 색상

색상(color)은 color 패키지 또는 xcolor 패키지를 이용한다. xcolor는 강력한 색상 제어 기능을 제공하므로 관련 문서를 반드시 한 번 읽어보기 바란다.

이 문서에서는 RGB 모델을 이용하여 색상을 몇 개 설정해서 사용하고 있다. 인쇄목적이라면 CMYK 모델이 더 적합하겠지만 화면용이라면 RGB로도 충분할 것이다.

```
%% 색정의.
\definecolor{shadecolor}{rgb}{0.80,0.82,0.75}
\definecolor{DarkMagenta}{rgb}{0.75,0.25,0.25}
\definecolor{CautionBlack}{rgb}{0.10,0.10,0.10}
\definecolor{TempColor}{rgb}{0.10,0.80,0.20}
```

shadecolor는 framed 패키지에서 사용하는 기본값으로서 shaded 환경의 배경색으로 쓰인다.

# 제 7 절 그밖의 몇 가지 문제

## 7.1 PSTricks 문제

PDFLaTeX의 경우이든 DVIPDFM$x$의 경우이든 pstricks를 전혀 지원하지 못하는 것은 pstricks 패키지를 즐겨 사용하는 사용자의 입장에서는 안타까운 일이다. 그러나 이 문제는 PDF 번역기 자체의 문제로서 현실적으로 단기간에 해결될 가능성은 높지 않아 보인다. 현재까지 개발된 PDF에서 PSTricks

사용하기 방법은 대강 몇 가지가 있다. 예를 들면 pdftricks나 ps4pdf가 그런 것들이다. 그러나 이 해결책들은 모두 PDFLaTeX을 위한 것이다. 이 분야에서는 DVIPDFM$x$가 취약함을 인정하지 않을 수 없다.

GhostScript를 직접 이용하지 않는 한 대부분의 해결책은 pstricks로 그린 그림을 잘라내어서 EPS 그림으로 만들고 이것을 일반적인 그림과 마찬가지로 취급하여 불러들이는 방법이다. 이 방법들의 대강에 대해서는 FAQ PSTricks 페이지를 참고하라.

pstricks의 부수 스타일인 pst-eps 패키지는 TeXtoEPS 환경을 제공한다. 원하는 그림을 이 환경 안에 넣고 컴파일한 다음 dvips(k)의 `-E` 옵션으로 변환하면 이 환경 안의 그림들을 EPS 파일들로 추출할 수 있다. 이 그림들을 원본 문서의 적절한 곳에 포함시킨다.

이런식으로 해결책을 잘 생각해서 활용해보면 아마도 간단한 pstricks 그림들은 어렵지 않게 처리할 수 있지 않을까 한다.

## 7.2 기울인 글꼴 문제

이 문서에서는 한글 사체(斜體)를 전혀 사용하지 않았다. 영문에서의 강조 부분에 대해서 이탤릭 글꼴을 쓰는 것은 하나의 관행이지만, 한글 글꼴에서는 이탤릭이란 있을 수 없으므로, 이탤릭이 주는 느낌과 유사하게 오른쪽으로 기울인(slanted) 글꼴을 거기에 대응시킨 것은 HLaTeX의 선택이었다. 반면, 선택의 여지가 있기는 했지만 hLaTeXp의 경우에는 `\emph`나 `\em` 명령이 있는 곳에 그래픽 글꼴을 대응시켰다.

조판상의 이유로 후자의 방법이 나을 것으로 생각한다. `\itshape`가 불렸을 때 한글 글꼴을 사체에서 다른 서체로 바꾸어주려면 `*.fd` 폰트정의파일을 수정하면 된다. 또는, 간단히 hitshape 패키지를 `\usepackage`하는 방법도 있다. 이 패키지는 한글에 대해서만 명조의 사체 대신 그래픽 글꼴을 사용하게 해준다.

## 7.3 밑줄 긋기

밑줄 긋기는 그다지 좋은 조판 관행은 아니다. 원래 이것은 원고를 타자기로 작성할 때 이탤릭체로 써야 할 곳을 표시하기 위해 쓰이던 도구였다. 그러나 이따금 밑줄을 그어야 할 때가 없는 것은 아닐 것이다.

`\underline` 명령은 비교적 잘 작동하지만 줄나눔이 되지 않는다. 밑줄 긋기에는 ulem 패키지가 흔히 쓰이는데, ulem이나 umoline이나 모두 한글 상황에서는 맞지 않아서 사용할 수 없었다.

myulem 패키지는[16] `\uline` 대신 `\huline`, `\sout` 대신 `\hsout` 등 h를 앞에 붙인 별도의 명령을 쓰도록 하고는 있지만 ulem 패키지의 기능을 대부분 제공한다. `[normalem]` 옵션도 작동한다. 다만, hlatex-interword 패키지와 함께 사용하는 경우, myulem 명령이 쓰인 범위 안에서는 자간 및 단어간격 설정이 작동하지 않는다.

## 7.4 상호참조 및 자동조사

LaTeX을 쓰는 즐거움 중의 하나가 상호참조와 인용을 자동으로 처리하는 것이 아닐까 한다. 그만큼 편리하고 좋은 기능이다. 한글 문서에서 상호참조 자체는 큰 문제가 없었다. 게다가 hLaTeXp와 HLaTeX은 상호참조로 불리워질 카운터의 호칭에 따라 자동으로 적절한 조사를 붙여주는 이른바 "자동조사"라는 탁월한 기능을 가지고 있다. 두 패키지 모두 hangul 패키지를 부르면 상호참조에 따르는 자동조사를 처리해주었다.

이 두 한글LaTeX 구현의 자동조사 루틴은 큰 차이가 없었다. `\label`이나 `\bibitem`과 같이 나중에 `\ref` 또는 `\pageref`, `\cite`로 불리워질 카운터에는 미리 표시를 해두었다가 나중에 실제로 불릴 때의 숫자를 확인하여 거기에 따라 조사를 붙여주는 기능이었다.

우리말의 조사 규칙은 표 1과 같다.

[16] 다른 h-시리즈 패키지들과는 달리 이 패키지만 my-로 시작하는 이름을 가지고 있는데, 특별한 이유가 있는 것은 아니고 원래 이 패키지가 지극히 개인적인 용도로 작성되었기 때문이다. hulem으로 이름을 바꾸기엔 너무 늦어 있었다. :-)

| 앞단어의 끝소리 | 와/과;을/를;이/가;은/는;라/이라 | 로/으로 |
|---|---|---|
| 리을 | 과;을;이;은;이라 | 로 |
| 리을 아닌 종성 | | 으로 |
| 중성 | 와;를;가;는;라 | 로 |

표 1: 우리말 조사 규칙

그런데, 이 자동조사 기능이 hyperref 패키지에서 문제를 일으켰다. 자동조사를 확인하기 위하여 삽입한 루틴이 hyperref의 label hyperlink 기능과 충돌한 것이다.[17]

그래서 hangul-nojosa가 만들어졌다. 이것은 HLaTeX 0.991의 hangul 패키지에서 조사 관련 기능만을 제거한 것이었다. 그 결과 PDF를 만드는 상황에서는 자동 조사 기능을 포기해야 하는 결과를 초래하였다.

이것은 엄청난 손실이었는데, 이제 hangul-k 패키지가 그 문제를 해결함으로써, 사실상 한글 PDF를 LaTeX으로 작성하는 것이 실질적으로 가능해졌다 할 것이다. 이 패키지는 김도현 님이 자동조사 구현의 새로운 아이디어를 제시한 후 며칠 동안 필자와 함께 구현되었다. 이로써 한글 PDF 문서를 어떤 희생도 없이 작성할 수 있는 길이 열린 것이다.

hangul-k에 관련된 자세한 내막은 부록 A를 보라.

다만, hyperref을 이용할 때 주의하여야 할 점은, label의 명칭은 반드시 영문으로 붙여야 한다는 점이다. 한글 label이나 인용 *key*에 대해서 hyperref이 에러를 내기 때문이다. 예컨대 `\label{그림1}`과 같은 한글 label은 허용하지 않는다. `\bibitem{아무개저자04}`와 같은 형식도 피하는 것이 좋다. hyperref 없이 한글 문서를 단지 hangul 패키지로만 작성하는 상황이라면 한글 label이나 한글 *cite_key*도 문제없다.

[17] 이 자동조사 루틴은 `*.aux` 보조파일에 자동조사 결정을 위한 정보를 집어넣었는데, 보조파일을 읽어서 처리하는 많은 다른 영문 스타일과도 충돌이 생기는 경우가 많았다.

## 7.5 완성형 밖의 한글 쓰기

이제 한글 PDF 문서 작성에서 마지막 남은 문제는 이른바 "완성형 제한"의 문제였다. Omega(Ω)나 Lambda(Λ)를 사용하지 않고 LaTeX을 쓰는 한, "완성형"이라 알려져 있는 EUC-KR 범위의 문자밖에는 조판할 수 없다는 것은 중대한 한계였다. 이 문제를 근본적으로 해결하는 길은 TeX의 8비트 제한을 넘어서는 Omega(Ω)를 채택하는 도리밖에 없어 보인다.

윈도 운영체제가 CP949 한글을 채택함으로써 사실상 윈도상에서는 현대의 모든 한글을 다 표현할 수 있게 된 것과 비교할 때 HLaTeX이 시대에 뒤져보인 이유 가운데 하나가 이 "완성형 제한"이었을 것이다.

KTUG에서는 완성형 표현의 범위를 넘어서는 한글 식자 방법에 대한 많은 토론과 기여가 이루어졌다. 그 결과, HLaTeX도 Lambda(Λ)를 이용하는 것이기는 하지만 CP949 한글을 식자할 수 있게 되었다. 그 후 조진환 님의 Omega-CJK, 김도현 님의 DHHangul과 같은 Omega(Ω)에 기반한 새로운 한글 문제 해결책들이 나오기 시작했다.

언젠가는 TeX에서의 한글 표현이 Omega(Ω)로 가게 될 것으로 믿는다. 그러나 LaTeX의 한계 내에서 PDF 한글 구현을 문제삼는 우리는 그 때까지 마냥 기다릴 수만은 없었다.

그래서 일종의 타협책으로 제시된 것이, CJK 패키지의 UTF8 환경을 이용하자는 것이었다. 이 환경은 유니코드의 UTF-8 엔코딩된 텍스트를 처리해 주는 기능이 있으므로, 이것을 이용해서 그다지 많지 않을 완성형 밖의 글자를 표현하는 데 쓰자는 발상이었다. 이래서 만들어진 것이 hlatexcjk 스타일 패키지이다. 이 패키지는 두 가지 명령과 한 개의 환경을 제공한다.

**\urchr 명령** 한 글자짜리 글자 파일을 만들어두고 이것을 불러들이는 명령이다. 예컨대 `ddom.char`라는 파일은 UTF-8 엔코딩된 "똠" 글자의 유니코드 값을 가지고 있다. 이것을 `\urchr{ddom}`으로 불러와서 식자하는 것이다.

**\UNI 명령** 김도현 님이 제안하신 명령으로, 특정 글자의 16진수 유니코드 값을 인자로 취하여 그 글자를 찍어준다.

**HCJK 환경** UTF-8 엔코딩으로 작성된 외부 텍스트 파일을 불러들이는 환경이다.

유니코드 사용에서 가장 문제가 되는 것은 역시 글꼴이다. 은글꼴은 모든 현대 한글을 다 표현할 수 있지만 중세문자가 없다.[18]

중세문자 문제는 매우 복잡한데, 관심있는 분은 FAQ 옛한글처리 페이지를 방문해보기 바란다.

hlatexcjk를 쓰게 되는 이유 중에는 "ᄒᆞᆫ글"과 같이 한두 글자의 고어를 사용해야 하는 경우가 많을 것이므로, 이럴 경우에 임시로 쓰게 하자는 것이 hlatexcjk의 제작목적이었기 때문에, 한컴바탕과 한컴돋움을 이용해서 그 글꼴에 내장된 완성형 중세문자를 그대로 가져다 쓰기로 하였다.[19]

실용적 목적으로 제작된 패키지이므로 꼭 사용할 필요가 있는 분은 사용해보시기 바란다. 다만, 크기가 만만찮은 HLaTeX과 CJK 패키지를 모두 불러들이다보니, 상당히 많은 메모리를 필요로 하는 점이 단점이다. 보통의 설정으로는 \urchr 명령으로 글자를 식자하는 데는 문제가 없겠지만 찾아보기를 만들면 메모리 부족을 호소할 지도 모른다. 이 글에서는 그래서 \urchr 명령이 쓰인 부분은 찾아보기에 포함하지 않았다.

[18] UnBatang-odal.ttf에는 중세문자 자소가 포함되어 있다. 그러나 일반적인 의미에서 고어 자소는 물론 완성형 고어를 포함하고 있지 않다고 해도 될 것이다.

[19] 이 방법은 일종의 편법이다. 중세문자 표현이 지향해야 할 바는 이렇게 글꼴 의존적으로 글꼴 자체의 사용자 영역에 심어져 있는 완성형 중세문자를 이용하는 방법이 아니라 "첫가끝" 방식이라 불리는 자소조합 형태로 구현되어야 온당하다고 생각한다.

## 제 8 절 마치는 말

TEX이든 LATEX이든 모두 글을 쓰기 위한 도구에 불과하다. 얼마나 훌륭한 문서를 만드느냐는 것은 도구의 성능에 달렸다기보다는 문서의 내용에 달린 것이 아닌가 한다. 단지 기술적으로 저자가 표현하고 싶은 것을 "표현할 수 있는 방법"이 있다는 것을 보여준 것으로 만족한다.

더 훌륭한 문서가 우리가 공들여서 마련한 다양한 패키지와 도구, 방법에 의하여 작성되고 공유되기를 희망한다. □

# 부록 A hangul-k 패키지에 관하여

hangul-k는 hangul-nojosa를 대체하는 새로운 한글 문서 작성용 LaTeX third-party 패키지이다. 이 패키지는 김도현 님이 "하이퍼링크-자동조사" 해결책을 제시함으로써 만들어졌는데, "하이퍼링크-자동조사"란, 기존의 자동조사 해결방법과는 달리 hyperref 패키지가 하이퍼링크를 다 만들기를 기다려서 거기에서 반환되는 문자열의 마지막 한두 글자를 분석하여 알맞은 조사를 붙여주는 획기적인 방식이다.

이 작업은 2004년 4월 말에서 5월 초에 집중적으로 이루어졌다. 최초의 아이디어를 제시한 글은 4월 28–29일에 올라왔고, 이것을 필자가 시험해본 다음 hangul-nojosa와 합치는 일을 하였다. 그것이 5월 1일의 일이었다. 그 뒤에 전개된 일의 간단한 요약이다.

① 김도현 님 : 두 문자를 취하여 검사할 수 있음을 보임.
② 필자 : 두 문자 반환루틴의 버그 지적
③ 김도현 님 : 개선된 두 문자 검사 루틴 제안
④ 필자 : `\if–\fi` 형식의 한글 및 기호문자 검사 루틴 제안
⑤ 김도현 님 : `josa.tab` 파일을 읽어서 일반적으로 모든 EUC-KR 문자를 검사할 수 있도록 하는 루틴 제안
⑥ 필자 : "비참조 자동조사" 기능을 추가
⑦ 김도현 님 : hyperref 패키지와의 인터페이스 개선

현재, hangul-k는 완전하지는 않다. 예컨대 `\cite` 명령이 숫자가 아닌 문자열을 되돌릴 때, 실제 "읽기"와 이 패키지가 선택하는 조사가 일치하지 않을 수 있다. 그리고 "비참조 자동조사" 기능은 편리하기는 하지만 "하이퍼링크 자동조사"와 충돌할 위험이 있다.

그러나, 현재의 상태로도 대부분의 문헌을 조판하는 것이 가능하고, 따라서 충분히 hangul 패키지를 적어도 PDF를 작성하는 목적으로는, 대체할 수 있다고 생각한다. 이것이야말로 지난 2년 넘게 고민해 온 문제가 해결된 셈이라 개인적으로는 정말 기쁜 일이었다.

# 부록 B 이 문서의 Preamble

이 문서의 Preamble 부분이다.

```
\documentclass[titlepage]{article}
%% PDF papersize setting.
\AtBeginDvi{\special{pdf: pagesize width 150mm height 212mm}}

%% hlatexcjk requires CJK and hangul-compatible packages.
\usepackage{CJK}
%% now, we use hangul-k style.
\usepackage{hangul-k}
%% "목차"를 "차례"로 바꿈.
\ksnamedef{contentsname}{차~례}

%% geometry를 이용한 layout 설정.
\usepackage[paperwidth=150mm,
    paperheight=212mm,
    hmargin=20mm,
    top=25mm,
    bottom=27mm]{geometry}
\setlength\headheight{27.5pt}

%% Hyperlinks and PDF Bookmarks.
\usepackage[dvipdfm,CJKbookmarks,colorlinks]{hyperref}
\AtBeginDvi{\special{pdf: tounicode KSCms-UHC-UCS2}}

%% fancyvrb for using Verbatim env.
\usepackage{fancyvrb} % before hlatexcjk

%% mflogo for using \MF
\usepackage{mflogo} % METAFONT logo.

%% hlatexcjk
\usepackage{hlatexcjk}

%% Font selection.
```

```
%% myttf는 makettfavailable의 예제 글꼴결합을 따라
%% ttf2hlatexfont로 개인적으로 조성된 글꼴 세트임.
\usepackage{myttf,bera}

%% 자간, 단어간격, 행간격.
\usepackage[default]{hlatex-interword}
\usepackage[hangul,adjustfootnotesep,adjustverbatim]{hsetspace}
\usepackage{hitshape} % 한글 \itshape의 그래픽글꼴 처리.
\usepackage[normalem]{myulem} % 한글 밑줄.

%\usepackage[ensec]{hsectsty}
\usepackage{hsectsty} % 한글 장절명령 스타일.
%% hsectsty 명령.
\sectionfont{\noindent\hfontseries{bc}\textcolor{blue}}
\subsectionfont{\noindent\hfontseries{bc}\textcolor{DarkMagenta}}
\subsubsectionfont{\noindent\hfontseries{bc}\sffamily\selectfont
  \textcolor{TempColor}}

%% Graphics, Colors, Frames
\usepackage{graphicx,color}
\usepackage{xcolor}
\usepackage{framed} % framed, shaded 환경.

%% 색정의.
%% shadecolor는 framed 패키지의 shaded 환경에서 사용함.
\definecolor{shadecolor}{rgb}{0.80,0.82,0.75}
\definecolor{DarkMagenta}{rgb}{0.75,0.25,0.25}
\definecolor{TempColor}{rgb}{0.10,0.80,0.20}

%%% 여러가지 정의들.
\usepackage{boxedminipage} % boxedminipage 환경.
%% MakeIndex
\usepackage{makeidx}
%% For Tabular
\usepackage{multirow,array,mdwtab}
%% list, compactlist
\usepackage{mdwlist}
```

```
\makecompactlist{enum**}{enumerate}
%% listing ㄴ
\usepackage{sverb}

%% Page headings
\usepackage{calc}
\usepackage{fancyhdr}
\pagestyle{fancy}
\lhead{\includegraphics[width=14mm]{ktugimage}}
\rhead{\colorbox{shadecolor}{\minipage{\linewidth - 17mm}%
    \raggedleft\sffamily\small\rightmark\qquad
    \thepage\endminipage}}
\chead{}\cfoot{}\rfoot{}\lfoot{}

%% LayOut Display
\usepackage{layouts}

%% Footnotes.
\usepackage[bottom]{footmisc}
\raggedbottom
%\usepackage{preview}

%% Author's commands and environments.
\newcommand\cntrdot{%
  \ifvmode\leavevmode\fi%
  $\,\cdot\,$%
}

\newcommand\bnm[1]{%
  \ifvmode\leavevmode\else\fi
  \kern -.45em『#1』\kern -.5em%
%%  위의 설정은 한컴바탕 글꼴용. 은글꼴은 커닝이 불필요하다.
%% 글꼴마다 이 부호의 커닝값이 다름.
%% 『#1』%
}
%% bookname 명령.
```

```
\newcommand\cmd[1]{%
  \index{명령!\textbackslash #1}\texttt{\textbackslash #1}%
}

\newcommand\pkg[1]{%
  \index{패키지!#1}\textsf{#1}%
}

\newcommand\file[1]{%
  \index{파일!#1}\texttt{#1}%
}

\newcommand\prgrm[1]{%
  \index{프로그램!#1}\textsf{#1}%
}

\newcommand\env[1]{%
  \index{환경!#1}\texttt{#1}%
}

\newcommand\teximp[1]{%
  \index{\THETeX!#1}\textrm{#1}%
}

\newcommand\THETeX{%
    \protect\TeX%
}
%% 이 이상한 명령을 두게 된 이유는 색인 만들기 때문.
%% HyperRef을 얹으면 \teximp 색인에서 \TeX 명령이 풀려버린다.

%% index 환경의 재정의
\makeatletter
\renewenvironment{theindex}
    {\if@twocolumn
       \@restonecolfalse
      \else
      \@restonecoltrue
```

```
      \fi
      \columnseprule \z@
      \columnsep 35\p@
      \twocolumn[\section*{\indexname}]%
        \@mkboth{\MakeUppercase\indexname}%
                {\MakeUppercase\indexname}%
        \thispagestyle{fancy}%
      \phantomsection
      \addcontentsline{toc}{section}{\protect\indexname}%
      \parindent\z@
      \parskip\z@ \@plus .3\p@\relax
      \let\item\@idxitem}
  {\if@restonecol\onecolumn\else\clearpage\fi}
\makeatother

%% maketitle
%% got from
%% http://zoonek.free.fr/LaTeX/LaTeX_samples_title/0.html
\makeatletter
\def\thickhrulefill{\leavevmode \leaders \hrule height 1pt%%
  \hfill\kern \z@}
\renewcommand{\maketitle}{\begin{titlepage}%
    \pagecolor{shadecolor}%
    \let\footnotesize\small
    \let\footnoterule\relax
    \parindent \z@
    \reset@font
    \null
    \vskip -5\p@
    \hbox{\mbox{%
        \hspace{-4pt}%
        \fbox{\includegraphics[width=3em]{ktugimage}}%
        \hspace{4pt}
        }%
      \vrule depth 0.9\textheight%
      \mbox{\hspace{2em}}
      \vtop{% %%%%%%%%%%%%%%%%%%
```

```
        \vskip 40\p@
        \begin{flushleft}
          \huline{\Large \@author}\par
        \end{flushleft}
        \vskip 80\p@
        \begin{flushleft}
          \huge \bfseries \@title \par
        \end{flushleft}
        \vfil
        }}
    \null
  \end{titlepage}%
  \pagecolor{white}%
  \setcounter{footnote}{0}%
}
\makeatother

%%% 92% 장평 한글 문자 사용을 위한 추가 정의.
%% Caption thinhangul setting.
\usepackage{ccaption}
\captiontitlefont{\normalfont\hfontseries{c}}
\captionnamefont{\normalfont\hfontseries{c}}

%% description label thinhangul setting.
\renewcommand\descriptionlabel[1]{%
    \hspace\labelsep
    \normalfont\hfontseries{c}\bfseries #1}

%% footnote thinhangul setting.
\let\origfootnotesize\footnotesize\relax
\def\footnotesize{\origfootnotesize\hfontseries{c}}

%% Parindent.
\setlength\parindent{1em}

%% eso-pic. cover.eps.
\usepackage{eso-pic}
```

```
\newcommand\CoverPicture{%
    \parbox[b][\paperheight]{\paperwidth}{%
        \vfill
        \centering
        \includegraphics[width=150mm,keepaspectratio]%
            {cover}%
%       \vfill
}}

%% Index
\makeindex

\newcommand\wi[1]{%
  #1\index{#1}%
}

%% Title page settings.
\title{아름다운 한글 PDF 문서 작성하는 법}
\author{도은이아빠}
\date{\today}

%% End of preamble. Now, start the document.
\begin{document}
```

# 찾아보기


TEX
  흔TEX, 7, 8
  DHHangul, 30
  HLATEX, 4, 7–9, 12, 18, 19, 23, 27–29, 31–33
  hLATEXp, 7, 8, 20, 27, 28
  Lambda(Λ), 3, 30
  MiKTEX, 9, 17
  Omega(Ω), 30
  Omega-CJK, 30
  PDFLATEX, 3, 4, 7, 12, 13, 15, 26, 27
  W32TEX, 17
c 시리즈 글꼴, 18, 19
가독성, 14, 18
각주, 19, 22, 23
강조, 27
고아와 과부(orphans and widows), 24
규격 용지, 14
그림 처리, 26
글꼴, 4
글꼴 내장, 6, 11
기울인 글꼴, 7
기울인(slanted) 글꼴, 27
길이(dimension), 21
김도현, 9, 10, 29–31, 33
단어간격, 20, 21
떠다니는 개체, 22
레이아웃, 14
마이너스 자간, 18
매킨토시, 6, 7
면주(面注), 24
명령
  startsection, 23
  \SetHangulspace, 22
  \UNI, 31
  \baselinestretch, 21, 23
  \bibitem, 28, 29
  \chapter, 14
  \cite, 28, 33
  \clubpenalty, 25
  \em, 27
  \emph, 11, 27
  \fi, 33
  \footnotesep, 23
  \hfontseries{c}, 19
  \hsout, 28
  \huline, 28
  \if, 33
  \interhchar, 21
  \interhword, 21

\itshape, 27
\label, 28, 29
\makethinhangul, 19
\pageref, 28
\raggedbottom, 25
\ref, 28
\rm, 11
\section, 14
\setstretch, 23
\sf, 11
\sout, 28
\textwidth, 14, 15
\tt, 11
\uline, 28
\underline, 28
\urchr, 30, 31
\usepackage, 27
\widowpenalty, 25
문장부호, 12
문화부 글꼴, 9
박원규, 9, 10
배경색, 26
본문글꼴, 11
비트맵 글꼴, 5, 6
사체(斜體), 27
산세리프글꼴, 11
상호참조, 28
색인, 19
수학 글꼴, 12
숫자, 12
신정식, 9
아시아폰트 기본팩, 9
안내문서, 3
암호, 3
여백, 14, 15
영문 글꼴, 12
온라인 문서, 3, 4, 14, 15
옵션 인자, 14
완성형, 30
워드 프로세서, 6, 7
윈도 운영체제, 6, 30
유니코드, 30
UTF-8, 30, 31
윤곽선 글꼴, 5
은광희, 8
은글꼴, 10
이탤릭, 11
이탤릭 글꼴, 27
인용, 28
자간, 19, 20
자동조사, 28, 29
장절명령, 19, 23
전자조판, 18
조진환, 9, 30
종이 규격, 14
책갈피, 3, 12–14

책갈피 만들지 않기, 14
출력소, 5
캡션, 19
커닝(kerning), 17
컴파일, 4, 13
타이포그라피, 19
텍스트 검색, 4
텍스트의 검색과 추출, 3
파일
  *.aux, 29
  *.bb, 25
  *.dtx, 20
  *.fd, 27
  *.ins, 20
  *.pfa, 5
  *.pfb, 5
  *.prn, 6
  *.ps, 6
  *.tfm, 18
  *.ttf, 18
  C:\texmf-asiafonts-basic, 9
  ddom.char, 30
  DVI, 3
  dvipdfmx.cfg, 9, 10
  EPS, 27
  hlatex-interword.dtx, 20
  josa.tab, 33
  JPG, 25
  map, 9, 10
  PDF, 25
  PK, 5, 7
  PNG, 25
  PS, 12
  UnBatang-odal.ttf, 31
  xxttf.sty, 19
파일네임 데이터베이스를 갱신, 9
판면, 15
패키지
  bmsec, 14
  CJK, 30, 31
  color, 26
  crop, 15
  fancyhdr, 24
  framed, 26
  geometry, 15, 17
  graphicx, 26
  hangul, 20, 23, 24, 28, 29, 33
  hangul-k, 3, 4, 29, 33
  hangul-nojosa, 29, 33
  hitshape, 27
  hlatex-interword, 3, 20, 28
  hlatexcjk, 30, 31
  hsectsty, 3, 23, 24
  hsetspace, 3, 22, 23
  hyperref, 12, 13, 17, 29,

33
layouts, 15
mskttfonts, 10
myulem, 3, 28
pdftricks, 27
ps4pdf, 27
pst-eps, 27
pstricks, 26, 27
sectsty, 23
setspace, 22, 23
size10d5.sty, 19
titlesec, 23
ulem, 28
umoline, 28
xcolor, 26
프로그램
흔글, 6, 7, 20
Acrobat Distiller, 6, 7
Acrobat Reader, 4, 5
Adobe Reader, 4, 11, 13
dvipdfm, 17
DVIPDFM$x$, 3, 4, 7, 10, 12–15, 17, 25–27
dvips(k), 7, 12, 27
ebb, 25
ezPDF, 6, 7
GhostScript, 7, 13, 25, 27
gv, 4
MiKTeX Options, 9
MS Word, 6
ps2pdf, 7, 13
QuarkXpress, 6
TTF2HLaTeXFont, 10, 11, 18, 19
ttf2pk, 7
xpdf, 4, 13
하이퍼링크, 4, 12, 13
한글, 12
한글 PDF, 6
한글 PDF 문서, 3
한글 문서, 3
해상도, 5
행간, 19
행간격, 22
화면보기, 3, 5
화면용 글꼴, 6
환경
HCJK, 31
quotation, 22, 23
quote, 22, 23
shaded, 26
spacing, 23
TeXtoEPS, 27
theindex, 19
titlepage, 19
UTF8, 30

verbatim, 22
활판 인쇄, 18

CMYK 모델, 26
Computer Modern 글꼴, 12
CP949, 30

double spacing, 21

EPS 그림파일, 25
EUC-KR, 30, 33

GPL, 8

KTUG, 3, 9, 13, 30
문서화 프로젝트 팀, 3
FAQ, 3

NFSS, 18

OpenType 글꼴, 6

PDF 그림, 25
PDF 변환 프로그램, 6
PDF에서 PSTricks 사용하기, 27
PostScript, 4
Preamble, 13

RGB 모델, 26

TrueType 글꼴, 5, 7
Type 1 글꼴, 5
Type 3 글꼴, 5

UHC 글꼴, 7, 8, 10