

# 레이텍을 빨리 쓸 수 있는 방법 2009

이호재\*

## 차례

1	참을성 없는 초보자들을 위한 해법 . . . . .	2
1.1	텍 배포판의 선택과 구입 . . . . .	3
1.2	설치 . . . . .	3
1.3	참고서 . . . . .	4
1.4	문제 해결 방법 . . . . .	5
2	반드시 레이텍이어야 하는가? . . . . .	5
3	올바른 레이텍 사용법 . . . . .	7
3.1	에디터에 의존하지 마라 . . . . .	7
3.2	텍 엔진에 대한 이해 . . . . .	7
3.3	문제가 생기지 않게 하려면 . . . . .	9
3.4	표를 쉽게 만들려면 . . . . .	9
3.5	그림이 떠다니지 않게 하려면 . . . . .	10
3.6	텍과 레이텍은 다르다 . . . . .	10

---

\*이 글은 2006년에 쓰여진 것이다. 최근 환경에 맞게 부분적으로 손질한 것은 김강수이다. 그러나 글의 논지는 변하지 않았다.

## 1 참을성 없는 초보자들을 위한 해법

이 글의 제목에 마음이 끌렸다면 당신은 분명 매우 절박한 처지에 있을 것이다. 하지만 당신은 참을성이 없기 때문에 —시간이 없다고 변명하겠지만— 이 책에서 만족할 만한 아무런 해법도 얻지 못할 수 있다. 이 글이 사기라는 비난을 할 수도 있겠지만 아마도 당신은 『142 분 동안 익히는 L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>』라는 책을 읽으면서 비슷한 경험을 이미 겪었기 때문에 그다지 크게 실망하지는 않을 것이다.<sup>1)</sup> 많은 사람들이 레이텍 입문서로 그 책을 권하지만 그 책을 보고 초보자가 당장 할 수 있는 일은 아무 것도 없다. 그 책은 당신에게 적합한 텍 배포판이 무엇이며 어디서 구할 수 있는지, 어떻게 설치해야 하는지, 결과물을 어떻게 얻어서 볼 수 있는지에 대해서는 설명하지 않는다.<sup>2)</sup> 그럼에도 그 책은 레이텍 입문자들에게는 여전히 가장 좋은 책이다. 그리고 믿기 어렵겠지만 그 책만으로도 —거의— 충분하다.

이 책의 목적은 당신이 불필요한 시행착오를 겪지 않도록 아주 신뢰할 만한 길잡이를 제시하는 것이다. 가장 좋은 길잡이는 단순하고 분명하다. 당신은 KTUG을 통해서 모든 것을 해결할 수 있다.<sup>3)</sup> KTUG에 관심이 없는 MiK<sub>T</sub>E<sub>X</sub> 사용자조차도 KTUG에서 조언을 얻을 수 있을 것이다. KTUG의 답변자들은 친절하고 풍부한 경험을 갖고 있다. 그들은 융통성 없고 게으른 질문자에게도 인내심을 갖고 성실히 답한다. 그들의 수고를 더는 방법은 간단하다.

1. KTUG Faq<sup>4)</sup>에서 해법을 검색한다. 여기에 답이 없다면 어디에서도 답을 얻기가 어려울 것이다. 하지만 이곳의 설명들은 초보자들이 이해하기에 다소 어렵다.
2. 해법을 찾을 수 없다면 KTUG 게시판에 질문하라. 반드시 문제를 자세히 설명하라. 아무리 노련한 의사라고 해도 환자가 증상을 얘기해주지 않으면 정확하게 진단할 수 없다. 어떤 패키지를 사용했는지, 오류 메시지는 무엇인지 설명하라. 가장 좋은 방법은 문제가 되는 파일을 첨부하는 것이다.

이제 성급한 당신의 수고를 덜기 위해 다음의 순서로 길잡이가 제시될 것이다.

1) <http://ftp.ktug.or.kr/mirrors/CTAN/info/lshort/korean/lshort-kr.pdf>. 흔히 lshort-kr이라 불리며, 명령행에서 `texdoc lshort-kr` 하면 바로 읽을 수 있다.

2) 컴파일하는 방법에 대한 간략한 설명이 있긴 하지만 대부분의 윈도우즈 사용자들은 그 설명을 제대로 이해하기가 어려울 것이다.

3) <http://www.ktug.or.kr/>

4) <http://faq.ktug.or.kr/faq/>

1. 텍 배포판의 선택과 구입
2. 텍 배포판의 설치
3. 참고서
4. 문제 해결 방법

### 1.1 텍 배포판의 선택과 구입

KTUG는 윈도우즈용 텍 배포판으로서 *ko.TeXLive*를 제공한다.<sup>5)</sup> 이것의 장점은 설치와 관리가 쉽다는 것이다. 당신이 유닉스 사용자라면…… 윈도우즈 사용자들은 유닉스 사용자들이 아주 똑똑하다고 생각한다. 그들은 유닉스 사용자들이 스스로 모든 것을 잘 해결한다고 생각한다. 나 역시 윈도우즈 사용자이다. 그래서 그들을 위해 내가 해줄 수 있는 말이라고는 리눅스 텍 시스템에 대해서도 KTUG이 조언을 해줄 수 있다는 것뿐이다.

종래 윈도우즈 TeX의 가장 강력한 대안은 MiKTeX이라 불리는 것이었다. 그러나 이 시스템에 대해서 KTUG은 거의 포기했다. 복잡한 이유가 여럿 있으나 간략히 말해서 KTUG의, 더 정확하게는 ko.TeX의 개발자들이 요구하는 기본적인 기능을 MiKTeX에서 얻을 수 없었다는 것이 가장 주된 이유이다. 만약 이 시스템을 설치하였다면, 그리고 한글 문서를 작성하여야 한다면, 가장 좋은 방법은 언인스톨하는 것이다.<sup>6)</sup>

### 1.2 설치

당신이 ko.TeXLive를 다운로드받았다면 텍 설치를 위해 당신이 할 일은 설치 프로그램을 실행하고 마우스를 두세 번 두드리는 것이다. ko.TeXLive는 KTUG이 2005년 이후 윈도우즈용 설치 프로그램을 제작해 온 경험이 충분히 녹아들어 있으므로 설치에 실패하거나 할 가능성이 거의 없다. 그래도 설치가 제대로 된 것 같지 않다면 재부팅해보거나 그래도 안 되면 다시 설치하라. 윈도우즈 사용자들은 대개 그렇게 하지 않는가?

5) 이것은 KC2008을 계승한 윈도우즈 텍 설치 프로그램이다.

6) 한글을 쓰지 않겠다고 했을 때 MiKTeX이 좋으나 TeXLive가 좋으냐는 논쟁은 무의미하다. 두 배포판이 다 탁월한 성능을 가지고 있으며 훌륭한 TeX 시스템으로 동작한다. 텍 시스템을 선택하는 기준은 단 한 가지뿐이다. 문제가 생겼을 때 해결하기 더 쉬운 것이 어느 것이냐는 것. 이 점에서 KTUG의 도움을 받기가 용이한 ko.TeXLive가 더 낫다고 말하는 것뿐이다. 만약 주변에 도움을 줄 일천이 충분하다면 굳이 ko.TeXLive를 선택할 필요도 없다. 그러나 이 글에서는 그런 경우를 고려하지 않았다.

전체(full) 설치와 기본(basic) 설치라는 두 종류의 인스톨러가 제공된다. ‘기본’ 설치라는 표현에 현혹되어서는 안 된다. 기본 설치 후에 벌어지는 모든 일은 사용자 자신의 책임이다. 기본 설치 후에 아무 것도 되는 것이 없다고 불평하는 것은 정당한 불평이 아니다. 텍 시스템에 대해서 정통하고 있다고 자부하지 않는다면 전체 설치를 선택하라.

텍 배포판은 글꼴, 컴파일러, 클래스, 패키지, 수 많은 텍 관련 프로그램 등으로 이루어진 거대한 시스템이다. 그러나 설치에 너무 많은 시간이 걸린다고 불평하지 마라. 실은 며칠씩 설치하고 설정해야 했던 예전에 비하면 그다지 많은 시간이 걸리는 것도 아니다. 길어야 30분에서 한 시간이면 충분하다.

### 1.3 참고서

이미 말하지 않았는가? 『142 분 동안 익히는 L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>』만으로 충분하다. *TeXBook*, *A Guide To LaTeX*, *The LaTeX Companion*, *TeX By Topic* 등 아주 많은 참고서들이 있다. 단언하건대 이 책들은 그다지 도움이 되지 않는다. 도움이 된다고 생각한다면 그건 착각이다. 언제가는 당신도 이 책들을 볼지도 모르지만 레이텍 디자이너가 되고 싶은 강렬한 욕구를 가지고 있지 않다면 참고서들은 잊어버려라. 만약 훌륭한 레이텍 디자이너가 되고 싶다면 많이 실험하라. 시행착오 만큼 좋은 스승은 없다. 도은이아빠는 latex.ltx 파일이 제일 큰 도움이 되었다고 말했다. 많은 공부와 실험과 시행착오가 latex.ltx가 좋은 교재가 되도록 했을 것이다. 다른 것과 마찬가지로 레이텍을 배우는 데에 왕도는 없다.

당신은 KTUG 게시판에서 당신이 원하는 것을 이루기 위해서 어떤 패키지를 쓰라는 말을 자주 듣게 될 것이다. 당신은 그 패키지가 어떤 것인지 좀 더 잘 알아두는 것이 좋다. 거의 모든 패키지에 설명서가 달려 온다. 대개 그 문서들의 파일 이름은 패키지가 이름과 같다. texdoc이 당신이 원하는 설명서를 찾아줄 것이다. 다음 명령을 명령행에서 실행하라.<sup>7)</sup> 명령행 실행이 낯설다면 texdoct라는 GUI 프로그램으로 원하는 문서를 찾아 읽을 수 있지만 이 경우 특히 한글과 관련된 모든 문서를 다 찾기는 어렵다. 또 한 가지 대안으로 KC2008Plus라는 보조 프로그램을 추가로 인스톨한다면 사용할 수 있는 KCmenu에서 좀더 쉽게 문서를 찾아

7) 레이텍을 제대로 사용하려면, 낯설겠지만 cmd 창을 이용하는 것이 좋다. 텍이 본래 명령행을 통해 이용하도록 만들어졌기 때문이다. ko.TeXLive의 기본 프론트엔드 프로그램인 texworks와 tlmgr은 문서의 작성/컴파일/수정/디버깅과 텍 시스템 자체의 유지/관리를 GUI 방식으로 할 수 있도록 도와준다. 심지어 KTUG은 KC2008에 있던 훌륭한 셸 환경인 KCmenu를 ko.TeXLive 2009와 함께 사용하는 방법까지도 제공하고 있다. 지나친 친절이라고 생각되는 면이 있을 정도이다.

읽는 것이 가능하지만 KC2008Plus를 설치할 것인지 여부는 선택사항이다. 따라서, 명령행을 열고 다음 명령을 써넣는 것이 가장 간단하다.

```
c:\>texdoc enumitem
```

중요한 것은, 꾸준히 문서를 읽으라는 것이다. L<sup>A</sup>T<sub>E</sub>X의 가장 큰 장점은 상상가능한 대부분의 문제에 대해 이미 해결책이 제시되어 있다는 것이다. 그것이 패키지들이다. 패키지에 대한 도움말은 패키지 문서에 정리되어 있다. 자신에게 필요한 패키지를 찾는 것은 게시판의 질문과 답변을 통할 수 있지만 그 패키지의 사용법은 문서를 끊임없이 읽으면서 적용해가는 수밖에 없다. 매뉴얼을 읽자!

#### 1.4 문제 해결 방법

레이텍의 기본 작동 원리를 이해하면 문제를 해결하기가 한결 수월해진다. 시행착오를 통해 그 원리를 스스로 터득하라. 이러한 방법과 고생은 반드시 필요하다. KTUG이나 직접 도움을 줄 수 있는 가까운 숙련자에게만 의지한다면 당신은 그다지 레이텍을 탐탁하게 생각하지 않는 사람일 것이다. 레이텍을 좋아하지도 않는데 레이텍을 잘 쓰게 되기를 기대할 수는 없다. 당신이 참으로 그런 사람이라면 워드 프로세서를 사용하거나 —적지 않은 비용이 들겠지만— 조판사에게 편집을 맡기는 것이 낫다.

당신이 참으로 레이텍을 아끼고 신뢰하는 사람이라면 문제 해결은 한결 수월해진다. 학회에서 제공하는 클래스가 문제를 일으킨다면 그 학회에 문의하라. 아니면 그 클래스를 사용해본 주변 사람에게 문의하라. 아마도 그들은 당신의 스승이거나 선배일 것이다. 만약 그들이 충분한 도움을 주지 못한다면 KTUG 게시판에 질문하라. 학회 클래스는 텍 배포판에 포함되지 않으므로 질문할 때 반드시 클래스 파일을 첨부하라.

## 2 만드시 레이텍이어야 하는가?

문제가 생기지 않게 하는 더 좋은 방법은 레이텍을 사용하지 않는 것이다. 피할 수 없는 강제된 사용이라면 생길 법한 일련의 문제에 대해서 당신은 책임지지 않아도 된다. 아마도 그 강제자가 당신의 모든 문제를 해결해줄 것이기 때문이다. 당신이 스스로 레이텍을 선택했다면 그것을 워드 프로세서처럼 사용할 수 있기를 바라지 마라. 차라리 워드 프로세서를 사용하라.

당신이 회사 내에서 편집과 조판을 맡고 있다면 고급 워드 프로세서 사용자가 되거나 고급 레이텍 사용자가 될 수 있을 것이다. 하지만 당신은 그런 일만 하지는 않을 것이다. 그런 일만 맡길 만큼 여유 있는 회사는 없다. 당신은 당신의 업무 중에서 조판의 비중을 어느 정도 뒤야 할지 그리고 그것을 어떻게 만족스러우면서도 손쉽게 처리할지 고민할 것이다. 적어도 레이텍을 조판 도구로 선택했다면 고민하지 마라. 당신은 조판 전문가가 아니므로 그것을 책임질 필요도 없고 잘 할 필요도 없다.

전자 출판 시스템은 제작을 아주 용이하게 했지만 대신 조판 규칙을 거의 완전히 무너뜨렸다. 글꼴을 쉽게 변경할 수 있다는 것이 처음에는 아주 편리하고 매력적인 기능으로 보였겠지만 지금 그것은 사람들이 제멋대로 문서를 만들게 했을 뿐만 아니라 거기에만 매달리게 만들었다.<sup>8)</sup> 도구가 도구로서 끝나지는 않는다. 당신은 동의하기 어렵겠지만 어떤 도구의 고유한 기능을—아마도 그것이 본래 의도였겠지만—이용하면 그것에 의해 서식이 결정되고 그 서식에 의해 당신이 작성하는 문서의 구조가 결정된다. 당신은 도구로부터 그다지 자유롭지 못하다. 레이텍을 사용해도 마찬가지이다. 레이텍의 가장 큰 장점 중의 하나가 저자가 내용에 충실할 수 있게 한다는 것이지만 레이텍에 익숙한 사용자들은 내용보다는 프로그래밍에 치중하기 십상이다. 이것은 초심자가 내용보다 모양에 치중하는 것보다 다를 바 없다.

레이텍은 다른 종류의 도구에 비해 훨씬 더 당신을 서식으로부터 자유롭게 할 것이다. 그것은 레이텍의 기능이 탁월해서가 아니라 서식에의 접근이 어렵기 때문이다. 익숙하지 않아서 이상하게 보이겠지만 레이텍의 기본 클래스나 패키지들이 만들어내는 서식이 사실 최선이다. 한국어 조판 규칙이 채 확립되지 않았고 영어 조판 규칙을 무차별적으로 끌어 쓰는 것이 바람직하지 않다는 지적이 있지만 적어도 그것이 차선책은 될 수 있다.

당신은 `\tableofcontents`, `\maketitle`, `\chapter`, `\section`, `\footnote` 명령만으로도 충분히 훌륭한 문서를 만들 수 있다. 당신이 만드시 레이텍이어야 한다고 생각한다면 그건 레이텍의 탁월한 기능들 때문이 아니라 문서에만 집중하겠다는 결심이 섰기 때문이다.

레이텍의 장점에 대한 수많은 논의가 있다. 그리고 그 논의의 수만큼 레이텍에 대해 비판하는 주장이 존재한다. 도구 자체의 장단점이란 사실 무익한 논란이다.

8) 이 말은 원래 워드 프로세서에만 해당했다. 그러나 X<sub>11</sub>TeX이 보편화되어 가고 있는 지금 TeX에서도 마찬가지로의 말을 할 수밖에 없게 되었다.

누가 레이텍을 쓰게 되는가? 입사한 회사의 문서 작성 시스템이 우연히도 레이텍 이거나, 자신이 선택한 지도교수가 우연히도 레이텍으로 논문을 쓰도록 강제(권유) 하는 경우이거나 등등. 사실 레이텍은 쓸 수밖에 없는 사람이 쓴다. 그런데 장단 점을 논해봐야 무엇하겠는가? 여기에 무슨 선교(mission)란 있을 수 없는 것이다. 자신이 선택한 도구의 특징을 이해하는 데, 그것의 장점을 스스로에게 납득시키는 만큼의 노력을 들여보라. 분명히 효과가 있을 것이다.

### 3 올바른 레이텍 사용법

#### 3.1 에디터에 의존하지 마라<sup>9)</sup>

가끔 WinEdt과 같은 에디터를 텍 그 자체와 혼동하는 경우를 본다. 특히 WinEdt은 좋지 않은데, 이 에디터가 유니코드를 제대로 지원하지 못하기 때문에, 한글을 많이 쓰는 사용자라면 이 에디터는 포기할 것을 권장한다. 텍 사용자들이 겪는 에디터 의존증은 피할 수 없는 면도 있지만, 그래도 이를 극복하는 쪽이 낫다. 편의성 측면에서는 부족하겠지만 `texworks`와 같은 에디터로도 얼마든지 훌륭한 문서를 작성할 수 있는 것이다.

에디터는 기호의 문제이므로 쉽게 바꾸기 어려울 것이다. 그러나 에디터에만 전적으로 의존하는 것은 극복하지 않으면 안된다. 가장 나쁜 것은 WinEdt에서 생긴 문제를 `TeX`의 문제라고 생각하는 경우이다. 이것은 자신의 코딩 실수를 `TeX`의 버그라고 주장하는 것과 똑같다. 대부분의 경우 실수의 원인은 자신이다. 이 점을 겸허하게 인정하자.

#### 3.2 텍 엔진에 대한 이해

텍을 1980년대에 만들어진 낡은 프로그램이라고 생각하는 것은 잘못이다. 불과 몇 년 전의 텍 사용방식과 지금은 상당히 많은 차이를 보여준다. 이 차이가 오래 전부터 텍을 사용해온 사람들이 곤잘 겪는 혼란의 원인이 된다. 예를 들면 이전에 보지 못하던 컴파일 루트, 이전에 표준이라고 생각했던 그림 포맷의 변화, 그리고 갑자기 사라진 `dvi`에 관련된 논의, 트루타입을 이용가능하게 해주던 예전에 존재 하던 패키지 묶음이 없어진 것 등이 그러하다.

9) 이 소절과 다음 절은 김강수가 추가하였음.

가끔  $X_{\text{g}}\text{TeX}$ 이니  $\text{Lua}\text{TeX}$ 이니  $\text{pdf}\text{TeX}$ 이니 하는 표현을 질문 답변 게시판에서 찾아볼 수 있을 것이다. 같은 레이텍인데도 여러 종류가 있는 것이다. `end user`의 컴파일 루트라는 점에서 각각의 명령 세트를 간략히 살펴보면 다음과 같다.

**latex+dvips+pspdf** 가장 전통적인 사용 방식이다. `pstricks`를 활용하는 데 기본적인 방법이지만 그림을 `eps`밖에 사용할 수 없고 결과(`pdf`)를 얻는 데 너무 오랜 시간이 걸린다는 점이 요즘 외면받는 주요한 이유가 되었다.

**latex+dvipdfmx** 일단 `dvi`를 생성하고 이를 `dvipdfmx`를 통하여 `pdf`로 변환하는 방법이다.  $\text{pdf}\text{TeX}$ 과 마찬가지로 최종 출력을 `pdf` 포맷으로 얻지만  $\text{pdf}\text{TeX}$ 보다 출력물의 크기가 작고 필요하다면 암호를 걸거나 하는 다른 기능을 추가적으로 가지고 있다.

**pdf $\text{TeX}$**  최근 가장 널리 사용되는 컴파일 루트이다. `pdflatex` 명령은 `dvi`를 만들지 않고 `pdf`로 출력한다. 엔진 자체에 `pdf`를 제어하는 여러 원시명령이 있으며 `micro-typography`라고 하는 고급 조판 기능을 제공한다.  $\text{pdf}\text{TeX}$ 에 특화되어 특별한 `pdf` 기능을 수행하는 패키지도 몇 가지 있다. `eps` 그림을 사용하는 것이 불가능하지 않지만 아무 준비없이 그냥 `eps`만 지시하면 문제가 생길 수 있어 권장하지 않는다. 그 대신 `pdf`, `png`, `jpg`와 같은 그림을 잘 처리한다.

**X $\text{g}\text{TeX}$**  유니코드 텍 엔진으로서 트루타입, 오픈타입 폰트를 자유롭게 이용할 수 있다. 최근  $\text{pdf}\text{TeX}$ 에서만 되던 일들이  $X_{\text{g}}\text{TeX}$ 에서도 되도록 포팅되는 일들이 급격하게 진행 중에 있다. `pstricks`도 거의 대부분의 기능이  $X_{\text{g}}\text{TeX}$ 에서 `compatible`하도록 변환되어 가고 있는 중이다. `dvi` 없이 `pdf`를 바로 생성한다.

**Lua $\text{TeX}$**  실험적인 개발중의 엔진으로서 스크립트 언어 `lua`, `metapost` 등 여러 기능을  $\text{TeX}$ 과 결합시킨 차세대 텍 엔진이다. `pdf`를 바로 생성한다.

보다시피, 이제 `dvi`는 `pdf`를 얻기 위한 중간 단계 정도의 의미밖에 갖지 않게 되었다.  $\text{TeX}$ 의 최종 출력은 `pdf`로 거의 표준화되어 있다. 그 이유는 `pdf`에 대해서 `inverse` 및 `forward search`가 가능해짐으로써 `dvi viewer`를 사용해야 할 필요가 적어졌기 때문이다. `latex+dvips+pspdf` 루트를 사용하는 경우에는 `pdf` 출력까지 가는 단계가 너무 길어 불편하므로 `dvi` 출력을 미리 보면서 작업하는 것이 효율적이 될 것으로 보이지만 실제 `pstricks` 코드를 출력 상태 그대로 100% 보여주는 `dvi previewer`란 존재할 수 없으므로 그다지 효율성을 확보하지 못한다. `ps viewer`



를 쓰면 되지 않느냐고 할지 모르지만 ps viewer 중에서 pdf viewer만큼 사용자를 만족시키는 것은 없고 ps viewer에서 스스로의 inverse search가 제대로 구현되어 있지 않다. pdf는 선택의 여지 없는 최종 출력물 포맷이다.

만약 아무런 제한도 없이 개인적인 용도로 사용하는 한글 문서를 작성한다면 이론의 여지 없이 Xe<sub>La</sub>T<sub>E</sub>X을 선택하는 것이 유리하다. 그 이유 중에서 제일 중요한 것은 한글 폰트의 사용에 제약이 없어진다는 바로 그 점이다. 그러나 학회나 학술지에 기고하는 경우, 그 쪽에서 요구하는 대로 맞추어야 할 때는 그에 따르는 것이 옳다.(대부분 pdf<sub>T</sub>E<sub>X</sub>을 표준으로 삼는 것이 요즘의 추세이다.)

예전 방식을 고집하는 것도 한 가지 선택이다. 그러나 그 책임은 자신이 져야지 새로운 텍 시스템에 돌려서는 안 된다. 초보자라 하더라도 선배나 동료에게서 물려받은 예전 파일들이 분명히 있을 것이다. 이런 파일들이 제대로 컴파일되지 않는다면 그것은 변화한 텍 환경에 적응하지 못한 때문이지 새로운 텍 시스템이 문제가 있기 때문이 아니다. 이 점을 거꾸로 생각하지 마라.

### 3.3 문제가 생기지 않게 하려면

문제를 해결하는 가장 좋은 방법은 문제가 생기지 않게 하는 것이다. 논문이 아닌 문서를 작성하는 경우에는 memoir 클래스 또는 oblvioir 클래스를 이용하는 것보다 현명한 활용 방법이 없다. 약간의 참을성만 갖고 memoir 설명서를 본다면<sup>10)</sup> KTUG에 질문할 일이 거의 없을 것이다.

### 3.4 표를 쉽게 만들려면

표를 쉽게 만들려면 표의 형식을 단순화하라. 당신이 표 안에 문장을 넣고 있다면 당신은 표를 오용하는 것이다. 표는 대등한 일련의 자료를 쉽게 비교할 수 있게 하기 위한 장치이다. 간격이 너무 좁아서 혼란을 일으키는 경우가 아니면 선은 필요하지 않다. 당신이 만드는 표들은 대개 목록으로 만들어졌어야 한다. 이것은 매우 잘못된 습관이다. 이런 습관의 원인은 단순하다. 당신은 문서 작성을 빨리 끝내고 싶어할 뿐이다. 표로 보여줄 수 있는 것은 생각보다 많지 않다. 당신이 참으로 당신의 독자를 아낀다면 표로 끝내지 말고 부지런히 더 설명하라.

10) texdoc memucs-manual.

### 3.5 그림이 떠다니지 않게 하려면

떠다니는 것이 무엇인지, 그리고 왜 떠다니는 것이 바람직한지 알면서도 그림이 떠다니지 않길 바란다면 당신은 분명 문서에 삽입해야 할 아주 많은 그림을 갖고 있을 것이다. 그렇다면 figure 환경을 이용하지 말고 caption 패키지의 \captionof와 같은 명령을 이용하라.<sup>11)</sup> 아니면 그냥 떠다니게 하라. 자상한 저자들은 독자를 배려하는 마음이 큰 나머지 갖가지 편의 장치를 마구 만들어 붙이고 싶은 충동에 휩싸이게 마련이다. 독자들은 명칭하지 않다. 그림의 위치나 글꼴의 크기가 조금 바뀌었다고 해서 내용에 질적 변화가 생기지도 않고 사용자가 오해하지도 않는다. 중요한 것은 당신이 그 그림에 대해 얼마나 잘 설명하느냐 하는 것이다.<sup>12)</sup>

### 3.6 텍과 레이텍은 다르다

텍이 레이텍의 근간을 이루고 있는 것은 분명하지만 텍과 레이텍은 다르다. 레이텍 디자이너가 되고 싶다면 텍을 잘 알아야 하겠지만 나는 당신이 그럴 생각을 갖고 있다고 믿지 않는다. 당신은 약간의 문제만을 갖고 있을 뿐이다. 그 문제를 해결하기 위해 당신이 구태여 텍 프로그래머가 될 필요는 없다. 부지런히 찾아보라. 당신과 같은 고민을 한 사람들이 많이 있고 이미 모든 해법이 다 나와 있다. 만약 해법이 없다면 당신이 원하는 것이 상식에서 너무 벗어나는 것이 아닌지 반성해 보라.

믿기 어렵겠지만 레이텍만으로도 당신이 원하는 거의 모든 것을 만들 수 있다. 레이텍은 간결하지 않지만 직관적이고 명료하다. 레이텍만 생각하라. 그러면 훨씬 더 편안하게 문서를 작성할 수 있을 것이다.

11) memoir (oblivoir) 클래스를 쓰는 것이 더 좋다.

12) 그림의 내용도, 대개 당신의 능력 밖에 있겠지만, 문장에 못지 않게 매우 중요하다. 언젠가는 이에 대해서도 논의할 기회가 생길 것이다.