The Asian Journal of TEX, Volume 1, No. 1, April 2007
Article revision 2007/03/02

LATEX 박스 매크로 분석

Dissection of LATEX Macros Related to Boxes

조진환 Jin-Hwan Cho
수원대학교 자연과학대학 수학과 chofchof@ktug.or.kr

ABSTRACT Boxes and glue are the key to understand how TEX makes complicated pages from individual characters and figures. Character boxes are put together into line boxes horizontally, and the line boxes are stacked vertically into paragraph boxes which finally compose page boxes. We analyze how LATEX macro commands related to boxes are composed of TEX primitives, and how the commands function. A few examples are prepared to show characteristics of the commands, and to focus the points to which ordinary LATEX users pay attention. The paper will be useful for LATEX users not only to make better LATEX documents but also to write their own style files.

1 들어가는 말

TEX이 하나의 판면을 조성하는 과정은 과거 구텐베르크에 의해 시작된 근대적 활판인쇄술[1]의 그것과 상당히 유사하다. 금속활자를 가지런히 놓아 하나의 줄을 만들고 다시 이러한 줄들을 세로로 나열해 하나의 판면을 만드는 과정은 TEX의 가장 중요한 개념인 박스와 글루를 이해하는 데 많은 도움을 준다.

TEX은 판면을 조성할 때 각 글자들의 모양을 고려하지 않는다. 대신 각 글자들을 하나의 사각 틀로 생각할 뿐, 글자의 모양이 이 틀 안에 들어있든 그렇지 않든 관심이 없다. 박스의 기본적인 형태는 바로 이런 사각 틀로 이루어져 있다. 사각 틀 안에 글자의 모양이 완전히 들어있지 않다는 점을 제외하면 TEX의 글자 박스들은 금속활자와 비슷하다. 또한 글자들로 만들어진 줄과, 이러한 줄들이 나열된 판면 자체도 박스의 일종이다.

TEX은 삼백여 개의 원시명령(primitive)들로 구성된 조판 시스템으로 프로그래밍이 가능하다는 점에서 다른 조판 시스템들과 큰 차이가 있다. 원시명령들을 조합해 사용하기 편리하도록 새로운 매크로(macro)들로 모아놓은 것을 포맷(format)이라 부른다. 대표적인 포맷으로는 TEX과 함께 발표된 Plain 포맷과 전세계에서 가장 많이 사용되는 LATEX[2] 포맷을 들 수 있는데, 포맷의 이름을 따서 각각 Plain TEX과 LATEX이라 부른다. 프로그래밍 언어로 비유하자면 TEX은 어셈블리, Plain TEX은 C, 그리고 LATEX은 객체지향 언어에 해당한다고 할 수 있다.

1. 근대적 활판인쇄술에 대한 자료는 [1]을 참조하자.
2. 이 글에서 LATEX은 1985년 Leslie Lamport에 의해 발표된 LATEX Version 2.09 대신 1994년 LATEX3 프로젝트 팀에 의해 발표된 LATEX2ε을 의미한다.

그림 1. 수평모드에서 글자 박스들로 만들어진 하나의 줄 박스

박스를 만드는 LaTeX 매크로들은 원시명령들의 조합으로 구성된다. 이 글의 목적은 이러한 구성을 분석해서 LaTeX 매크로들이 어떠한 원리로 박스를 만드는지, 그리고 어떠한 방법으로 매크로들을 활용할 수 있는지 여러가지 예제를 통해 살펴보는 것이다. LaTeX 사용자들은 원리를 이해함으로써 보다 다양한 방법으로 문서를 만들 수 있고, 특히 자신의 매크로 패키지를 만들고자 하는 경우 이 글에서 다루게 될 내용들이 여러모로 도움을 줄 것이다.

2 TeX의 모드와 글루

TeX은 크게 세 가지 모드(mode)에서 박스를 구성한다.[3] 여러 개의 글자 박스들을 가로로 나열해 줄 박스를 만드는 방식을 수평모드(horizontal mode)라 부르고, 이러한 줄 박스들을 세로로 나열해 큰 판면 박스로 구성하는 방식을 수직모드(vertical mode)라 부른다. 또 한가지 방식은 수식으로 이루어진 박스를 만드는 수식모드(math mode)로 이 글에서는 다루지 않는다.

수평모드에서 하나의 줄 박스를 구성할 때 글자들로 이루어진 기본 박스뿐만 아니라 다른 형태의 박스들도 사용된다. 내부 수직모드에서 만들어진 여러 줄로 이루어진 박스는 물론 수식모드에서 만들어진 수식 박스도 하나의 박스로 간주해서 수평모드에서 쓸 수 있다는 것이다. 그림 1은 수평모드에서 글자 박스들로 어떻게 줄 박스를 만드는지 보여준다. 이 그림에서 'TeX'이라는 단어는 다른 단어들과 달리 글자 박스들이 서로 겹쳐있다. TeX은 이러한 효과를 어떻게 처리할까? 단어 사이의 공백(space), 그리고 수평모드에서 박스의 좌우 이동을 다룰 때 TeX은 글루를 사용한다.[4]

글루는 간단히 폭이 정해진 공백으로 생각할 수 있다. 박스를 왼쪽으로 이동하는 효과는 폭을 음수로 지정함으로써 얻는다. 이러한 기본적인 특징이외에 다른 조판 시스템에서 찾기 힘든 글루의 특징은 폭이 상황에 따라 늘어나거나 줄어들 수 있다는 것이다. 일반적

3. TeX에는 모두 여섯 가지 모드가 있다. 수평모드는 글자 박스들을 가로로 나열해 하나의 줄 박스를 만드는 것으로 이 줄 박스는 나중에 여러 개의 줄 박스로 나뉘게 된다. 또 다른 수평모드는 제한된(restricted) 수평모드로 줄 바꿈을 허용하지 않고 하나의 줄을 만드는 것으로 주로 `\hbox` 원시명령이 박스를 만들 때 나타난다.

수직모드는 줄 박스들을 세로로 나열해 하나의 판면 박스를 구성하는 것으로 이 박스는 나중에 여러 개의 판면 박스들로 나뉜다는 점에서 수평모드와 비슷하다. 또한 제한적 수평모드와 비슷한 역할을 하는 것으로 내부(internal) 수직모드가 있다. 이 모드는 주로 `\vbox` 원시명령이 박스를 만들 때 나타난다.

나머지 두 가지 방식은 수식모드와 디스플레이(display) 수식모드이다. 수평 및 수직모드에 대한 자세한 설명은 [2, Chapter 6]와 [3, Chapter 24, 25]에 잘 나와있다.

4. 엄밀히 말해 그림 1에서 'TeX'을 이루는 글자 박스들은 ⟨glue⟩가 아니라 ⟨dimen⟩에 의해 이동되었다. 이 글에서 '글루'는 ⟨dimen⟩과 ⟨glue⟩를 모두 지칭하는 용어로 사용된다. 글루에 대한 자세한 설명은 [2, Chapter 8]와 [3, Chapter 12]에서 찾을 수 있다.

으로 폭이 고정된 글루는 ⟨dimen⟩으로 표시하고 그렇지 않은 글루는 ⟨glue⟩로 구별해서 표시한다. 예를 들어 1cm는 ⟨dimen⟩을, 그리고 `1cm plus 0.5cm minus 0.5cm`는 ⟨glue⟩를 나타내는데 그 의미는 기본 폭 1cm에서 0.5cm까지 늘어나거나 줄어들 수 있다는 뜻이다. 즉, 최대 1.5cm에서 최소 0.5cm까지 폭이 변하는 공백이다.

단순한 ⟨dimen⟩대신 ⟨glue⟩를 사용하는 가장 큰 목적은 최적의 줄 바꿈을 얻기 위함이다. 일반적으로 TEX은 ⟨glue⟩에서 줄 바꿈을 허용하지만 ⟨dimen⟩에서는 그렇지 않다. 좀 더 엄밀히 말하면 ⟨glue⟩를 주는 TEX 원시명령 `\hskip`에서는 줄 바꿈이 허용되지만 ⟨dimen⟩을 주는 원시명령 `\kern`은 바로 다음에 `\hskip`이 나타나지 않는 한 줄 바꿈이 허용되지 않는다. 원시명령 `\hskip`에는 ⟨glue⟩뿐만 아니라 ⟨dimen⟩도 사용할 수 있다. 이 경우 ⟨dimen⟩은 가변 정보가 `plus 0pt minus 0pt`인 ⟨glue⟩로 대치된다. 마찬가지로 ⟨glue⟩를 `\kern` 명령에 사용할 경우 가변 정보가 사라진 ⟨dimen⟩으로 바뀐다. 수직모드에서는 원시명령 `\vskip`이 `\hskip`의 역할을 하고 `\kern`은 수평 및 수직모드에서 동일하게 사용할 수 있다.

3 수평모드에서 박스를 만드는 LATEX 매크로

제한된 수평모드에서 하나의 줄 박스를 만드는 대표적인 원시명령은 `\hbox`이다. TEX 사용자들이 한번 쯤 경험하는 실수 중 하나는 예제 1과 같이 `\hbox` 명령을 문단의 처음에 사용하는 것이다. 아마 세 줄이 출력됨을 예상하겠지만 결과는 오른쪽과 같이 모두 네 줄이 출력된다. 왼쪽 코드에서 첫 줄과 셋째 줄에 대한 결과는 예상과 같을 것이다. 왜 둘째 줄은 한 줄에 나타나지 않고 두 줄에 걸쳐 출력되는가?[5]

```
1  The\par
2  \hbox{Korean} \TeX{}\par
3  Society
```

The
Korean
TEX
Society

예제 1. `\hbox`를 문단의 처음에 사용한 경우

예제 1의 코드에서 앞 절에서 다룬 TEX의 모드가 어떻게 변하는지 살펴보자. 먼저 첫 줄의 결과 'The'는 수평모드에서 세 글자 박스들을 가로로 조합해 얻는다. 첫 줄의 마지막에 나오는 원시명령 `\par`는 수평모드를 끝내고 (지금까지 조성된 줄 박스를 여러 개의 줄 박스로 나눈 후) 수직모드로 전환하는 역할을 한다. 즉, 하나의 문단이 끝난 것이다.

둘째 줄 처음에 나오는 `\hbox`는 'Korean'이라는 단어를 제한된 수평모드에서 하나의 박스로 만든다. 하지만 `\hbox`가 수직모드를 수평모드로 전환하지 않는다는 점에 유의하자. 수평모드로 전환되는 시점은 다음에 나오는 `\TeX`이라는 매크로를 만날 때이다.[6] 이제 'TEX'이라는 단어는 수평모드에서 조성되고 `\par` 명령을 만나면서 하나의 박스가 된다.

5. 이 결과를 제대로 예측한 독자라면 아마  기호(dangerous bend)의 의미를 알고 있을 것이다.

6. `\TeX` 매크로는 `latex.ltx` (1327줄)에 `\def\TeX{T\kern-.1667em\lower.5ex\hbox{E}\kern-.125emX\@}`로 정의되어 있다. 이 매크로는 글자 'T'에서 시작하므로 이 때 수평모드로 전환된다.

다시 모드는 수직으로 바뀌고 마지막 줄에서 'Society'를 포함한 박스가 수평모드에서 만들어진다. 이 과정에서 수직모드에서 세로로 배열할 박스의 개수는 모두 네 개가 된다. 예제 1의 결과를 얻는 이유가 바로 여기에 있다.

3.1 \hbox를 대체하는 LaTeX 매크로 \mbox

원시명령 \hbox는 제한된 수평모드에서 하나의 박스를 만들기 때문에 TeX 사용자들은 \hbox 자체가 항상 수평모드에서 동작한다고 잘못 생각하기 쉽다. 이 명령이 주로 수평모드에서 나타나는 것이 사실이지만 수직모드에서 나타날 수도 있다. LaTeX은 이러한 혼동을 피하고자 항상 수평모드에서 동작하는 새로운 매크로 \mbox를 제공한다.

```
\mbox{내용}
```

이제 \mbox가 어떻게 정의되어 있는지 latex.ltx에서 살펴보자.[7] 코드의 첫머리에 나오는 숫자들은 파일 내의 줄 번호를 의미한다.

```
4562  \long\def\mbox#1{\leavevmode\hbox{#1}}
```

정의에 따르면 \mbox는 단지 \hbox에 \leavevmode를 덧붙인 것이다. \leavevmode가 어떤 역할을 하는지 알기 위해 다시 latex.ltx를 찾아보자.

```
492  \def\leavevmode{\unhbox\voidb@x}
```

먼저 \voidb@x는 아직 아무것도 지정되어 있지 않은 박스 레지스터를 의미하고, 원시명령 \unhbox는 박스 레지스터에 저장된 \hbox의 내용을 부리고(unload) 그 내용을 지우는 역할을 한다.[8] 그렇다면 아무 것도 없는 박스를 부리는 \leavevmode는 무엇을 하는 것일까? \leavevmode의 핵심적인 역할은 박스를 부리는 것이 아니라 수직에서 수평으로 모드를 바꾸는 것이다. \hbox 명령은 이 역할을 하지 않았지만 \unhbox 명령은 이 역할을 한다 [3, p. 283]. 따라서 수평모드에서는 아무 것도 없는 박스를 부림으로써 아무런 영향을 주지 않는 반면, 수직모드에서는 문단을 시작하는 수평모드로 바꾸는 효과를 얻는다. 이제 예제 1의 코드에서 \hbox를 \mbox로 바꾼 예제 2를 실행해보자.

```
4  The\par
5  \mbox{Korean} \TeX{}\par
6  Society
```

The
Korean TeX
Society

예제 2. \mbox를 문단의 처음에 사용한 경우

7. 이 글은 LaTeX2ε (2005/12/01) 버전을 다룬다. 파일 latex.ltx는 $TEXMF/tex/latex/base/ 폴더에서 찾을 수 있다.

8. 원시명령 \unhbox (내용을 지우지 않으려면 \unhcopy)는 박스의 내용을 부리는 반면 \box (내용을 지우지 않으려면 \copy)는 박스 자체를 사용한다는 점이 다르다. 예를 들어 {\setbox0=\hbox{a b c d e f}\vbox{\hsize=1cm\copy0\unhbox0}}를 실행하면 \copy0이 만든 결과는 줄 바꿈이 일어나지 않는데 반해 \unhbox0이 만든 결과는 줄 바꿈이 생긴다.

워드프로세서에서 return 또는 enter 키를 반복해서 입력하면 여러 개의 빈 줄을 얻는다. 앞에서 설명한 \leavevmode를 이용하면 LaTeX에서도 같은 효과를 낼 수 있다. \leavevmode\par를 반복해서 입력해보라. 하지만 TeX에서는 수직모드용 글루[9]를 이용하는 것이 더 좋은 결과를 얻는다.

3.2 \hbox to 및 \hbox spread 이상의 효과를 얻는 LaTeX 매크로 \makebox

원시명령 \hbox에는 박스의 폭을 조절해 주는 \hbox to와 \hbox spread 같은 변화형이 있다. 예제 3은 이러한 명령들에 의해 어떻게 박스의 폭이 변화하는지 보여준다.

```
\let\B\textbar
\B\hbox{Korean \TeX}\B\par
\B\hbox to 30mm{Korean \TeX}\B\par
\B\hbox spread 10mm{Korean \TeX}\B
```

```
|Korean TeX|
|Korean      TeX|
|Korean    TeX|
```

예제 3. \hbox to와 \hbox spread를 이용한 박스 크기 조절

하지만 \mbox의 정의에는 \hbox가 고정되어 있으므로 이와 같은 변화형을 사용할 수 없다. 이를 위해 LaTeX은 \hbox의 변화형보다 편리하고 다양한 기능을 가진 \makebox 매크로를 제공한다.

```
\makebox[박스폭][정렬방법]{내용}
```

예제 4는 \makebox를 이용해 예제 3을 다시 처리한 것이다.

```
\B\makebox{Korean \TeX}\B\par
\B\makebox[30mm][s]{Korean \TeX}\B\par
\B\makebox[\width+10mm][s]{Korean \TeX}\B
```

```
|Korean TeX|
|Korean      TeX|
|Korean    TeX|
```

예제 4. \makebox를 이용한 박스 크기 조절[10]

매크로 \makebox는 latex.ltx에서 다음과 같이 정의된다.

```
\def\makebox{%
  \leavevmode
  \@ifnextchar(%)
    \@makepicbox
    {\@ifnextchar[\@makebox\mbox}}
```

이 매크로도 \mbox와 같이 \leavevmode로 시작하므로 항상 수평모드에서 출발한다. 다음으로 LaTeX 매크로 \@ifnextchar는 세 개의 인자를 받는데 다음에 오는 글자가 첫째 인자와 같다면 둘째 인자를 실행하고 그렇지 않다면 셋째 인자를 실행한다 (latex.ltx

9. 수직모드에서 ⟨glue⟩를 주는 원시명령은 \vskip이고 ⟨dimen⟩을 주는 것은 수평모드와 동일하게 \kern이다. LaTeX에서는 보다 많은 기능을 가지고 있는 \vspace 매크로 또는 \bigskip, \medskip, \smallskip을 주로 쓴다. [4, p. 35] 또는 [5, p. 857]를 참조하라.

10. 13째 줄과 같이 사칙연산을 사용하려면 먼저 calc.sty 패키지를 불러야 한다.

796줄). 따라서 \makebox 바로 뒤에 오는 글자가 '('인 경우 \@makepicbox를, '['인 경우 \@makebox를, 그리고 어느 쪽도 아닌 경우 \mbox를 수행한다. 아무 옵션도 없는 \makebox{#1}은 바로 \mbox{#1}임을 알 수 있다.

매크로 \@makepicbox는 내부 수직모드에서 박스를 만드는 \vbox 원시명령을 사용하므로 이 절에서는 \@makebox 매크로만 분석하기로 한다. 다음 코드는 TEX에서 옵션을 받아들이는 전형적인 코딩 방법이다.

```
\def\@makebox[#1]{%
  \@ifnextchar [{\@imakebox[#1]}{\@imakebox[#1][c]}}
```

매크로 \@makebox는 하나 또는 두 개의 인자를 가지는데 둘째 인자가 주어지지 않을 경우 [c]를 둘째 인자의 기본값(default)으로 설정해서 새로운 매크로 \@imakebox를 부른다.

```
\long\def\@imakebox[#1][#2]#3{%
  \@begin@tempboxa\hbox{#3}%
    \setlength\@tempdima{#1}%       support calc
    \hb@xt@\@tempdima{\csname bm@#2\endcsname}%
  \@end@tempboxa}
```

매크로 \@imakebox는 세 개의 인자를 가지는데 처음 두 인자는 \@makebox에서 전해진 것으로 순서대로 박스의 폭과 박스 내에서 내용의 정렬 방법을 의미한다. 셋째 인자에는 \makebox에서 건네진 박스의 내용이 들어간다. 두 번째 인자는 [c|l|r|s] 네 가지 옵션을 받는데 [c|l|r]은 각각 가운데, 왼쪽, 그리고 오른쪽 정렬을 의미하고 [s]는 양쪽 정렬을 의미한다.[11]

맨 처음에 수행하는 매크로 \@begin@tempboxa는 두 개의 인자를 가지는데 첫째 인자에는 \hbox가 들어가고 둘째 인자에는 박스의 내용이 들어간다. 이 매크로의 역할은 주어진 내용을 레지스터 \@tempboxa에 저장한 후, 이 박스의 폭(\width), 높이(\height), 깊이(\depth), 그리고 높이와 깊이를 더한 전체높이(\totalheight)를 측정하는 것이다. 정의는 다음과 같다.

```
\long\def\@begin@tempboxa#1#2{%
   \begingroup
     \setbox\@tempboxa#1{\color@begingroup#2\color@endgroup}%
     \def\width{\wd\@tempboxa}%
     \def\height{\ht\@tempboxa}%
     \def\depth{\dp\@tempboxa}%
     \let\totalheight\@ovri
```

11. latex.ltx 4575째 줄에서 이 정보를 얻을 수 있다. 박스에 들어갈 내용은 이미 \@begin@tempboxa 매크로에서 레지스터 \@tempboxa에 저장되어 있다. [c] 옵션은 \hss\unhbox\@tempboxa\hss 방법으로 내용을 부린다. 양쪽에 \hss가 붙어있으므로 가운데 정렬의 효과가 나타난다. ⟨glue⟩를 주는 원시명령 \hss에 대한 자세한 설명은 [3, p. 71]을 참조하라.

또한 [l] 옵션은 \unhbox\@tempboxa\hss와 같이 오른쪽에만 \hss가 붙어있으므로 오른쪽이 끝없이 줄거나 늘어나는 효과를 얻는다. 반대로 [r] 옵션은 \hss\unhbox\@tempboxa와 같이 왼쪽에만 \hss가 붙어있다. 마지막으로 [s] 옵션은 양쪽에 \hss를 붙이지 않고 \@tempboxa를 부리는 역할만 하지만 4582째 줄에서 \hb@xt@, 즉 \hbox to를 사용하므로 (각주 12 참조) 양쪽 정렬의 효과가 나타난다.

```
4572        \totalheight\height
4573        \advance\totalheight\depth}
4574    \let\@end@tempboxa\endgroup
```

레지스터 \@tempboxa에 들어있는 내용과 박스 크기에 대한 정보는 \begingroup과 \@end@tempboxa 매크로, 즉 \endgroup으로 둘러싸여 있으므로 \@imakebox 매크로 정의의 4581째 줄 및 4582째 줄에서만 유효하다. 하지만 4582째 줄의 둘째 인자 #2는 [c|l|r|s] 네 개의 옵션만 허용하므로 4581째 줄의 박스 폭을 설정하는 첫째 인자 #1에서만 사용할 수 있다.

4581째 줄에서 \setlength가 붙은 이유는 예제 4의 13째 줄과 같이 calc.sty 패키지에서 지원하는 사칙 연산을 사용하기 위해서이다. 마지막으로 4582째 줄의 \hb@xt@는 \hbox to와 같으므로[12] 첫째 인자에서 주어진 박스의 폭만큼 공간을 확보한 다음, 이 안에 둘째 인자에서 주어진 정렬 방법으로 레지스터 \@tempboxa에 있는 (셋째 인자에 의해 주어진) 내용을 부린다.

3.3 \makebox 매크로의 활용 하나

박스에 들어갈 내용의 크기 정보를 알고 있다는 것은 유용하다. 예를 들어 박스의 폭을 두 배로 늘린 후 내용을 넣는 방법은 예제 5와 같다. 가운데 정렬이 기본값임을 기억하자.

```
14  \B\makebox[2\width]{Korean \TeX}\B\par
15  \B\makebox[2\width][l]{Korean \TeX}\B\par
16  \B\makebox[2\width][r]{Korean \TeX}\B\par
17  \B\makebox[2\width][s]{Korean \TeX}\B
```

```
|      Korean TEX      |
|Korean TEX            |
|            Korean TEX|
|Korean          TEX   |
```

예제 5. \makebox를 이용한 박스 폭 늘리기

2대1 혹은 1대2 정렬과 같이 다른 정렬 방법을 원할 경우에는 정렬에 \hss를 사용하지 않는 [s] 옵션이 유용하다 (각주 11 참조). 예제 6을 보자.

```
18  \B\makebox[30mm][s]{A\hfil\hfil B\hfil C}\B\par
19  \B\makebox[30mm][s]{A\hfil B\hfil\hfil C}\B\par
```

```
|A            B      C|
|A       B           C|
```

예제 6. \makebox를 이용한 2대1 및 1대2 정렬

또한 \rlap 및 \llap[13] 매크로도 각각 \makebox[0pt][l] 및 \makebox[0pt][r]로 구현할 수 있다. 예제 7에서 보듯 \rlap과 \llap은 문단 처음에 사용할 경우 예제 1과 같은 현상이 발생하는 데 반해 \makebox[0pt][l|r]을 쓰면 항상 새로운 문단을 시작하는 효과를 얻는다. \makebox[0pt][l|r]을 적절히 이용하면 다른 재미있는 결과도 얻을 수 있다.[14]

두 매크로 \mbox와 \makebox로 만든 박스에 테두리를 주려면 \fbox와 \framebox를 대신 쓰면 된다. 옵션이나 사용법은 동일하다.

12. \def\hb@xt@{\hbox to} (latex.ltx 566째 줄)
13. \def\rlap#1{\hb@xt@\z@{#1\hss}} (latex.ltx 4847째 줄), \def\llap#1{\hb@xt@\z@{\hss#1}}

```
\rlap{\B K\B}orean
\par
\llap{\B K\B}orean
\par
\makebox[0pt][l]{\B K\B}orean\par
\makebox[0pt][r]{\B K\B}orean\par
\makebox[0pt]{\B K\B}orean
```

```
|K|
orean
|K|
orean
|Korean
|K|orean
|Korean
```

예제 7. \rlap, \llap 및 \makebox[0pt]의 활용

```
\fbox{내용}
\framebox[박스폭][정렬방법]{내용}
```

예제 8과 같이 테두리의 굵기는 \fboxrule로 조절하고 테두리와 내용의 간격(padding)은 \fboxsep으로 조절한다.[15]

```
\fbox{Korean \TeX}\par
\framebox[2\width]{Korean \TeX}\par
\setlength\fboxrule{2\fboxrule}
\framebox[2\width][l]{Korean \TeX}\par
\setlength\fboxsep{2\fboxsep}
\framebox[2\width][r]{Korean \TeX}
```

Korean TEX

Korean TEX

Korean TEX

Korean TEX

예제 8. 박스에 테두리를 주는 \fbox와 \framebox 매크로

3.4 \makebox 매크로의 활용 둘

LATEX에서 \makebox(#1,#2)[#3]{#4} 형태의 매크로는 picture 환경에서 주로 쓰이지만 picture 환경 바깥에서도 이것을 이용하면 다양한 효과를 얻을 수 있다.

```
\makebox(박스폭,박스높이)[정렬방법]{내용}
```

이 매크로는 세 가지 옵션을 받는데 처음 두 옵션 (#1,#2)는 각각 박스의 폭과 높이를 설정하는 데 쓰인다. 이 때 주의할 것은 앞의 예제들과 달리 \width나 \height 등의 박스 정보를 쓸 수 없다는 것이다. 또한 20mm와 같이 길이를 직접 주는 것이 아니라 20과 같이 숫자로 주어야만 한다. 이 경우 20은 \unitlength[16]에 설정된 값의 20배를 의미한다. 세 번째 옵션 [#3]은 박스 내용의 정렬 방법을 결정하는데 세로 정렬은 [t|b]로, 가로 정렬은 [l|r|s]로 준다. 가로 및 세로 모두 지정하지 않으면 가운데 정렬이 기본값이다. 박스에 테두리를 주려면 예제 9와 같이 \framebox를 대신 사용한다.

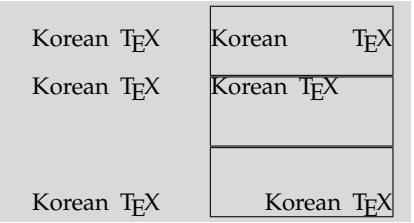
(latex.ltx 4848째 줄)

14. \rlap 또는 \llap을 이용하면 overfull 경고 없이 주어진 범위를 벗어난 곳에 글을 쓸 수 있다. 예를 들어 코드의 첫머리에 나오는 줄 번호들은 모두 판면을 벗어난 여백에 위치하고 있다. 이 글에서 코드 출력을 위해 사용한 moreverb.sty 패키지의 listing 환경은 \llap을 이용한다. 또한 \rlap은 두 글자를 겹쳐 출력하는 경우에 사용된다. 예를 들어 \rlap{$\times$}$+$는 +와 ×가 겹친 ∗를 출력한다.

15. 테두리 간격의 기본값은 3pt이고 테두리 굵기의 기본값은 0.4pt이다. (latex.ltx 7757–7758째 줄)

16. \unitlength는 기본값이 1pt로 설정되어있다. (latex.ltx 5200째 줄)

```
\setlength\unitlength{2mm}
\makebox(12,4.5){Korean \TeX}\quad
\framebox(12,4.5)[s]{Korean \TeX}\par
\makebox(12,4.5)[t]{Korean \TeX}\quad
\framebox(12,4.5)[tl]{Korean \TeX}\par
\makebox(12,4.5)[b]{Korean \TeX}\quad
\framebox(12,4.5)[br]{Korean \TeX}
```



예제 9. picture 환경에서 주로 쓰는 \makebox와 \framebox

4 수평모드에서 박스를 위아래 이동하는 LaTeX 매크로

TEX은 제한된 수평모드에서 조성된 박스를 위아래로 이동하는 두 원시명령 \raise와 \lower를 가지고 있다. 이 명령들은 오직 수평모드에서만 사용할 수 있다. 즉, \par 다음에 \hbox 명령은 나올 수 있지만 \raise나 \lower는 나올 수 없다. 예제 10에서 \raise 및 \lower 명령이 적용된 후 박스의 높이와 깊이가 어떻게 변하는지 기억하자.

```
K\raise 5pt\hbox{O}R\lower 5pt\hbox{E}A
\fbox{K}\fbox{\raise 5pt\hbox{O}}\fbox{R}%
        \fbox{\lower 5pt\hbox{E}}\fbox{A}
```

예제 10. \raise와 \lower를 이용한 박스의 위아래 이동

LaTeX도 수평모드에서 박스를 위아래로 이동하는 매크로 \raisebox를 제공한다.

```
\raisebox{이동폭}[박스높이][박스깊이]{내용}
```

원시명령 \raise가 \hbox 명령으로 조성된 박스만 인자로 받아들이는 데 반해 매크로 \raisebox는 \makebox나 \framebox처럼 박스에 들어갈 내용도 인자로 받아들인다. 이 매크로도 수직모드에서 사용할 경우 수평모드로 전환한 다음 박스를 조성한다. 매크로 정의는 다음과 같다.

```
\def\raisebox#1{%
  \leavevmode
  \@ifnextchar[{\@rsbox{#1}}{\@irsbox{#1}[]}}
\def\@rsbox#1[#2]{%
  \@ifnextchar[{\@iirsbox{#1}[#2]}{\@irsbox{#1}[#2]}}
```

\raisebox는 최소 두 개에서 최대 네 개의 인자를 받아들인다. 항상 첫 인자에는 박스의 위아래 이동 폭이 들어가고 마지막 인자에는 박스의 내용이 들어간다. 나머지 인자들은 옵션으로 사용되는데 첫째 옵션은 박스의 높이를, 둘째 옵션은 박스의 깊이를 재설정하는 데 사용된다. 만약 옵션이 주어지지 않으면 박스의 높이와 깊이는 변하지 않는다. 위의 코드는 인자의 수에 따라 다음과 같이 새로운 매크로로 대치한다.

```
\raisebox{#1}{#2}           ⇒  \@irsbox{#1}[]{#2}
\raisebox{#1}[#2]{#3}       ⇒  \@irsbox{#1}[#2]{#3}
\raisebox{#1}[#2][#3]{#4}   ⇒  \@iirsbox{#1}[#2][#3]{#4}
```

먼저 \@irsbox 매크로가 어떤 역할을 하는지 살펴보자.

```
4823 \long\def\@irsbox#1[#2]#3{%
4824   \@begin@tempboxa\hbox{#3}%
4825     \setlength\@tempdima{#1}%
4826     \ifx\\#2\\\else\setlength\@tempdimb{#2}\fi
4827     \setbox\@tempboxa\hbox{\raise\@tempdima\box\@tempboxa}%
4828     \ifx\\#2\\\else\ht\@tempboxa\@tempdimb\fi
4829     \box\@tempboxa
4830   \@end@tempboxa}
```

이미 \makebox 매크로 분석에서 본 것과 같이 \@begin@tempboxa는 주어진 내용 #3을 레지스터 \@tempboxa에 저장한 후 박스의 크기 정보를 측정한다. 4825째 줄에서 위아래 이동 폭 #1을 임시 레지스터 \@tempdima에 저장하고, 옵션 [#2]가 주어진 경우 또 다른 임시 레지스터 \@tempdimb에 저장한다. 4827째 줄은 레지스터 \@tempboxa에 저장된 박스를 \@tempdima에 저장된 길이만큼 \raise 명령을 이용해 올린 다음 이 결과를 다시 \@tempboxa 레지스터에 저장한다. 이제 옵션 [#2]가 주어진 경우 4828째 줄에서 레지스터 \@tempboxa의 높이가 \@tempboxb에 저장된 값으로 재조정된다. 마지막으로 4829째 줄에서 \@tempboxa에 저장된 박스가 출력된다. 즉, #3의 내용을 박스로 만든 후 #1에서 주어진 값만큼 올린 다음 #2가 주어진 경우 이 값으로 높이를 재조정하는 것이다. 이 경우에도 박스의 크기 정보(\width, \height, \depth, \totalheight)는 두 매크로 \@begin@tempboxa와 \@end@tempboxa 사이에서 유효하므로 위아래 이동 폭을 주는 #1과 재설정할 높이 값을 주는 #2에만 쓸 수 있다.

또 다른 매크로 \@iirsbox도 같은 역할을 하는데 #2에서 주어진 값으로 높이를 재조정한 후 (4837째 줄) #3에서 주어진 값으로 깊이도 재조정한다 (4838째 줄).

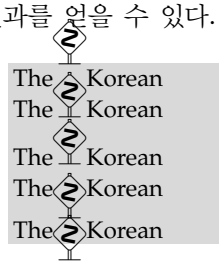
```
4831 \long\def\@iirsbox#1[#2][#3]#4{%
4832   \@begin@tempboxa\hbox{#4}%
4833     \setlength\@tempdima{#1}%
4834     \setlength\@tempdimb{#2}%
4835     \setlength\dimen@{#3}%
4836     \setbox\@tempboxa\hbox{\raise\@tempdima\box\@tempboxa}%
4837     \ht\@tempboxa\@tempdimb
4838     \dp\@tempboxa\dimen@
4839     \box\@tempboxa
4840   \@end@tempboxa}
```

이렇게 높이와 깊이를 재조정함으로써 어떤 효과를 얻을 수 있을까? 각주 5를 보면가 본문과 각주를 가르는 선 위까지 올라와 있다.[17] 이 효과는 \raisebox 매크로의 높이 재설정 옵션을 0pt로 지정함으로써 얻을 수 있다. 예제 11을 보면 처음 두 줄은 \textdbend와 상관없이 줄 간격이 일정하다. 수평모드에서 특정 글자 또는 박스 때문에 줄 간격이 변화하는 것을 막고자 할 때 이와 같은 트릭을 쓸 수 있다. 넷째 줄도 마찬가지로 높이를 0pt로 재설정하는 옵션이 들어갔지만 줄 간격이 고르지 않다. 이것은

17. 이 줄의 \textdbend도 각주 5와 같은 효과를 주어 윗 줄과 겹친다. \textdbend 매크로는 manfnt.sty 패키지를 불러야 쓸 수 있다.

\textdbend가 전체높이(\totalheight)의 반만큼 아래로 이동해서 깊이가 그만큼 길어졌기 때문이다. 마지막 줄처럼 깊이마저 0pt로 재설정하면 원하는 결과를 얻을 수 있다.

```
The\raisebox{.5\totalheight}[0pt]{\textdbend}Korean\par
The\raisebox{0pt}[0pt]{\textdbend}Korean\par
The\textdbend Korean\par
The\raisebox{-.5\totalheight}[0pt]{\textdbend}Korean\par
The\raisebox{-.5\totalheight}[0pt][0pt]{\textdbend}Korean
```



예제 11. 높이와 깊이 재설정에 따른 줄 간격의 변화

TEX의 조판 방식은 글자 박스를 이용해 먼저 줄 박스를 만든 후 이것을 이용해 판면 박스를 만드는 것이다. 따라서 판면의 특정한 위치에 무엇을 임의로 넣는 것은 쉽지 않을 뿐더러 TEX이 원하는 조판 방식에도 어긋난다. 굳이 이런 작업을 해야만 하는 경우에도 어떤 것을 넣을 위치가 절대위치(absolute position)라면[18] 판면 조성을 시작하는 곳(원점)에서만 가능할 뿐 다른 곳에서는 불가능하다. 왜냐하면 일반적인 TEX의 조판 과정에서 현재 위치를 아는 것은 불가능하기 때문이다.[19]

그렇다면 어떤 것을 넣을 위치가 현재 위치에 대한 상대위치(relative position)로 주어진다면 어떨까? 현재 위치에서 위로 0.5cm, 왼쪽으로 2cm 정도 떨어진 곳에  를 넣으려면 \makebox와 \raisebox 매크로를 예제 12와 같이 이용해보자.

```
A% 현재 위치; 가로이동 후 세로이동
\raisebox{5mm}[0pt][0pt]{%
  \makebox[0pt][r]{\textdbend\hspace{2cm}}}%
B\par C% 현재 위치; 세로이동 후 가로이동
\makebox[0pt][l]{\hspace{2cm}%
  \raisebox{-1cm}[0pt][0pt]{\textdbend}}%
D
```

AB
CD

예제 12. 특정 위치에 임의의 박스 넣기

5 수평모드에서 박스를 저장하는 LATEX 매크로

앞 절의 \makebox 및 \raisebox 매크로들은 모두 주어진 박스 내용의 크기를 측정하기 위해 \setbox 원시명령으로 박스에 들어갈 내용을 \@tempboxa라는 박스 레지스터에 저

18. 절대위치는 미리 설정된 판면의 원점을 중심으로 위치를 계산하는 것을 말한다. 포스트스크립트(PostScript)나 PDF의 경우 판면 좌측 하단을 원점으로 설정하고 직교좌표계(Cartesian coordinate)를 사용하는 데 반해 TEX 및 DVI는 판면 좌측 상단을 원점으로 설정하고 y-축의 방향이 뒤바뀐 직교좌표계를 사용한다. 줄 박스들을 이용해 판면 박스를 만들 때 위에서 아래로 나열하지 아래에서 위로 나열하지 않는다는 점을 고려하면 왜 이러한 좌표계를 사용하는지 이해할 만하다.

19. pdfTEX은 현재 위치를 알려주는 원시명령 \pdflastxpos와 \pdflastypos를 제공한다. 하지만 이 명령들도 판면 박스를 출력하는 마지막 과정에서만 제대로 사용할 수 있다. 따라서 이 정보를 이용하려면 같은 문서에 대해 적어도 두 번 TEX 작업을 해야 한다. 첫 번째 작업 때 외부 파일에 위치 정보를 기록한 후, 두 번째 작업 때 이 정보를 읽은 후 사용하면 된다.

장했다는 것을 기억하자. 이 절은 TEX에서 박스에 들어갈 내용을 레지스터에 저장하고 나중에 다시 부르는 과정을 다룬다.

Knuth가 만든 원래의 TEX은 모두 256개의 박스 레지스터를 가지고 있다.[20] 각주 8에서 이미 언급한 대로 박스 레지스터에 내용을 저장하는 원시명령은 \setbox, 박스를 사용하는 원시명령은 \box, 그리고 박스를 부리는 원시명령은 \unhbox이다. 만약 번호가 0인 박스 레지스터에 'A'라는 글자를 저장하려면 \setbox0=\hbox{A}를 실행하고, 이 박스 레지스터를 사용하려면 \box0, 부리려면 \unhbox0을 실행하면 된다. 특별히 마지막 255번 박스 레지스터는 판면 조성을 위해 특별히 사용되므로 조심해야 한다.

사용자들이 박스 레지스터의 번호를 일일이 기억하는 것은 결코 쉽지 않다. 이를 위해 plain TEX은 번호 대신 이름을 사용할 수 있도록 주어진 이름에 번호를 할당하는 매크로 \newbox를 제공한다. 예를 들어 \newbox\mybox를 실행하면 \mybox라는 이름에 특정한 번호가 할당되고 \setbox\mybox=\hbox{A}와 같이 사용할 수 있다.

LATEX은 \newbox 이외에 추가로 \newsavebox라는 매크로를 제공한다. 이 매크로는 이미 번호가 할당된 이름에 다시 번호를 할당하려고 할 경우 에러를 발생하는 기능을 가지고 있다. 그리고 \hbox를 확장한 \mbox와 \makebox가 LATEX에 있듯이 원시명령 \setbox를 확장한 \sbox와 \savebox도 있다.

```
\newsavebox{레지스터}
\sbox{레지스터}{내용}
\savebox{레지스터}[박스폭][정렬방법]{내용}
```

```
4612  \long\def\sbox#1#2{\setbox#1\hbox{%
4613    \color@setgroup#2\color@endgroup}}
```

먼저 \sbox의 정의를 살펴보자. 이 매크로는 두 원시명령 \setbox와 \hbox를 모두 포함하고 있기 때문에 \setbox와 달리 곧바로 박스의 내용을 받을 수 있다. 예를 들어 \setbox\mybox=\hbox{A} 대신 \sbox\mybox{A}와 같이 편리하게 쓸 수 있다. 또 다른 매크로 \savebox의 정의는 \makebox와 거의 유사하고 동일한 옵션 및 사용법을 가진다.

```
\usebox{레지스터}
```

```
4635  \def\usebox#1{\leavevmode\copy #1\relax}
```

박스 레지스터에 저장된 내용을 사용하는 LATEX 매크로는 \usebox로 위의 정의에서 보듯 사용 후 레지스터의 내용을 지우는 원시명령 \box 대신 지우지 않는 \copy를 이용한다. 따라서 원하는 만큼 반복해서 쓸 수 있다. 예제 13을 통해 \usebox 매크로와 \copy 및 \unhcopy 원시명령의 차이를 살펴보자.

20. TEX이 일반에 처음 발표된 1980년대 초는 8비트 컴퓨터가 주종을 이루는 시대였기 때문에 $256 = 2^8$개의 박스 레지스터를 가지는 것은 자연스러운 일이었다. 하지만 시간이 지남에 따라 보다 많은 박스 레지스터를 요구하게 되었고, 현재 거의 표준으로 사용되고 있는 (pdf)ε-TEX의 경우 $32768 = 2^{15}$개의 박스 레지스터를 사용할 수 있다. TEX의 16비트 확장판인 Omega의 경우 이보다 많은 $65536 = 2^{16}$개의 박스 레지스터를 제공한다.

```
55  \newsavebox\mybox
56  \sbox\mybox{A B C }
57  \newcommand\UB{\usebox\mybox}
58  \newcommand\BC{\copy\mybox}
59  \newcommand\UC{\unhcopy\mybox}
60  \UB\UB\UB\UB\par
61  \BC\BC\BC\BC\par
62  \UC\UC\UC\UC
```

A B C A B C A B C A B C
A B C
A B C
A B C
A B C
A B C A B C A B C A B C

예제 13. \usebox 매크로와 \copy 및 \unhcopy 원시명령의 차이

예제 13에서 박스 레지스터 \mybox에 저장된 내용은 끝에 공백을 가지고 있음에 유의하자. 오른쪽 첫 줄은 코드 60째 줄의 실행 결과로 \mbox{A␣B␣C␣}를 연속해서 네 번 입력한 것과 동일한 결과를 준다. 따라서 줄 바꿈이 일어나지 않았다. 오른쪽 둘째 줄부터 다섯째 줄까지는 코드 61째 줄의 실행 결과이다. 이러한 현상이 발생하는 것은 이미 예제 1에서 본 이유와 동일하다. 즉, \copy도 \hbox와 마찬가지로 수직에서 수평으로 모드를 전환하지 않는다. 마지막 두 줄은 코드 62째 줄의 실행 결과로 \usebox를 실행한 결과와 달리 줄 바꿈이 일어났다. \unhcopy 명령이 박스를 사용하는 것이 아니라 박스의 내용을 부리기 때문에 'A␣B␣C␣A␣B␣C␣A␣B␣C␣A␣B␣C␣'를 입력한 것과 같으므로 이 같은 줄 바꿈 효과를 얻는 것이다.

LaTeX이 가진 큰 특징 중 하나는 \begin{환경이름}으로 시작해서 \end{환경이름}으로 끝나는 논리적인 환경(environment)을 가진다는 것이다. 박스에 내용을 저장할 때에도 이와 같은 환경을 사용할 수 있도록 LaTeX은 lrbox 환경을 제공한다.

```
\begin{lrbox}{레지스터} 내용 \end{lrbox}
```

예제 14에서 myenv는 환경 안에 있는 내용을 박스 레지스터 \mybox에 저장한 다음 이것을 세 줄에 걸쳐 출력한다. 단, 마지막 줄은 \raisebox 매크로를 이용해 첫 줄과 둘째 줄 사이로 올림으로써 마지막 줄이 빈 효과를 준다. 특히 문단 박스를 구성하는 명령들과 함께 사용하면 훨씬 다양한 효과를 얻을 수 있다. [5, p. 870]의 boxedminipage 환경은 lrbox 및 minipage 환경을 함께 사용한 좋은 예제이다.

```
63  \newenvironment{myenv}
64    {\begin{lrbox}{\mybox}} % 환경 시작 때 수행
65    {\end{lrbox}%           % 환경 끝날 때 수행
66     \usebox\mybox\par\usebox\mybox\par
67     \raisebox{1.5\baselineskip}[0pt]{\usebox\mybox}}
68  \begin{myenv}The Korean \TeX{} Society\end{myenv}
69  \begin{myenv}The Asian Journal of \TeX\end{myenv}
```

The Korean TEX Society
The Korean TEX Society
The Korean TEX Society

The Asian Journal of TEX
The Asian Journal of TEX
The Asian Journal of TEX

예제 14. lrbox 환경을 이용한 사용자 환경 정의

박스 레지스터에 저장된 박스의 크기 정보는 \wd, \ht 및 \dp 원시명령으로 추출할 수 있을 뿐만 아니라 바꿀 수도 있다. 하지만 박스의 내용 자체가 변하는 것이 아니라 내용을 둘러싸고 있는 박스의 크기만 변한다는 점을 기억하자. 예제 15를 보라.

```
\sbox\mybox{Korean}
\the\wd\mybox\par
\usebox\mybox\usebox\mybox\par
\setlength{\wd\mybox}{2\wd\mybox}
\the\wd\mybox\par
\usebox\mybox\usebox\mybox
```

```
28.88972pt
KoreanKorean
57.77945pt
Korean          Korean
```

예제 15. 박스의 크기 정보 변경

6 수직모드에서 박스를 만드는 LaTeX 명령

수직모드에도 수평모드에서 박스를 조성할 때 사용했던 여러 원시명령들과 비슷한 기능을 하는 것들이 있다. 제한된 수평모드에서 박스를 만드는 원시명령 \hbox에 해당하는 것은 \vbox와 \vtop이다. 이 두 원시명령은 내부 수직모드에서 주어진 박스들을 세로로 배열해 하나의 박스로 만든다. 이 때 세로 길이는 주어진 박스들이 얼마나 많으냐에 따라 달라지지만 가로 길이는 주로 \hsize 원시명령에 설정된 값으로 결정된다. 이 값을 건드리지 않았다면 판면의 가로 길이와 같다. 원시명령 \vbox가 만든 박스는 마지막 줄에 정렬점(reference point)을 가지는데 반해 \vtop이 만든 박스는 첫 줄에 정렬점을 가진다. 예제 16을 통해 확인하자.

```
% 현재 \hsize는 색칠된 면의 가로 폭이다.
A\vbox{B\par C}D\vtop{E\par F}G\par
A\vbox{\hbox{B}\par\hbox{C}}D%
 \vtop{\hbox{E}\par\hbox{F}}G\par
A\vbox{\mbox{B}\par\mbox{C}}D%
 \vtop{\mbox{E}\par\mbox{F}}G
```

```
 B
AC      DE      G
         F

 B
ACDEG
    F
 B
AC      DE      G
         F
```

예제 16. 원시명령 \vbox와 \vtop의 차이

예제 16의 77째 줄을 모드의 변화에 따라 분석해 보자. 글자 'A'로 시작하기 때문에 수평모드에서 줄 박스를 만든다. 이 줄 박스는 다섯 개의 박스가 가로로 배열되는데, 처음은 'A'를 포함한 글자 박스, 두 번째는 \vbox가 만든 박스, 세 번째는 다시 'D'를 포함한 글자 박스, 네 번째는 \vtop이 만든 박스, 그리고 마지막은 'G'를 포함한 글자 박스이다.

이제 \vbox가 내부 수직모드에서 어떻게 박스를 만드는지 살펴보자. 먼저 글자 'B'를 만나면서 내부 수직모드는 줄 바꿈이 가능한 수평모드로 전환된다. 따라서 글자 'B'를 포함한 박스는 \hsize에 설정된 값을 폭으로 가진다. 즉, \vbox가 만드는 박스는 폭이 \hsize로 주어진 두 개의 박스를 세로로 배열한 것이다. 같은 방법으로 \vtop도 박스를 만드는데 \vbox와 달리 정렬점을 글자 'F'에 두지 않고 글자 'E'에 둔다. 위의 코드를 \hsize 변화없이 그냥 수행하면 글자 'D'부터 판면의 바깥에 위치하기 때문에 보이지 않을 수 있다.

내부 수직모드에서 수평모드가 나오지 않으면 어떻게 될까? \hbox는 제한된 수평모

드에서 박스를 만들지만 수직모드를 수평모드로 바꾸지 않는다. 따라서 \hsize에 설정된 값이 폭으로 사용되지 않으므로 77째 줄이 만든 박스와 다른 형태의 박스가 만들어진다.

이번에는 \hbox를 모두 \mbox로 바꾸어보자. 어떤 예상을 했는가? 매크로 \mbox도 제한적 수평모드에서 박스를 만든다는 점은 \hbox와 같지만 이미 앞 절에서 보았듯이 \mbox는 항상 수직모드를 수평모드로 바꾼다. 따라서 77째 줄과 같이 \hsize에 설정된 값이 폭으로 사용되므로 처음과 같은 결과를 얻는다.

원시명령 \hbox에 \hbox to와 \hbox spread 같은 변화형이 있듯이 \vbox와 \vtop도 동일한 변화형을 가진다. 수평모드에서는 박스를 위아래로 이동하기 위해 \raise와 \lower 원시명령을 사용했다. 마찬가지로 수직모드에서는 박스를 좌우 이동하기 위해 원시명령 \moveleft와 \moveright를 사용한다. \vbox와 \vtop으로 만든 박스의 폭이 어떻게 결정되는지는 위에서 설명했다. 하지만 높이와 깊이를 결정하는 규칙은 그리 간단하지 않다. 자세한 규칙은 [3, pp. 80–81]을 참조하라.

LaTeX은 내부 수직모드에서 박스를 구성하는 매크로 \parbox와 보다 많은 확장성을 제공하는 minipage 환경을 제공한다.

6.1 문단 박스를 만드는 LaTeX 매크로 \parbox

매크로 \parbox는 최소 두 개에서 최대 다섯 개의 인자를 받아들인다.

```
\parbox[정렬점위치][박스높이][내부정렬]{박스폭}{내용}
```

마지막에 주어지는 두 인자들 중 앞 인자에는 박스의 가로 폭이 들어가고 뒤 인자에는 박스의 내용이 들어간다. 나머지 인자들은 옵션으로 사용되는데 첫째 옵션에는 [c|b|t]가 올 수 있고 이에 따라 박스의 정렬점이 각각 가운데, 마지막 줄, 첫 줄에 위치한다. 둘째 옵션은 박스의 높이를 지정하는 데 쓰인다. 그리고 셋째 옵션은 박스 내부의 세로 정렬 방법을 의미하는데 [c|t|b|s]가 올 수 있다. 이 옵션들의 의미는 가로가 세로로 바뀌었다는 것을 제외하면 \makebox에서 보았던 것과 완전히 동일하다.[21] 예제 17은 이러한 옵션들이 변함에 따라 \parbox가 만드는 박스의 모양이 어떻게 변하는지 보여준다. 만약 [s] 옵션에 따른 변화를 보려면 82째 줄에서 \par를 \vss로 바꾼 후 실행해 보자.

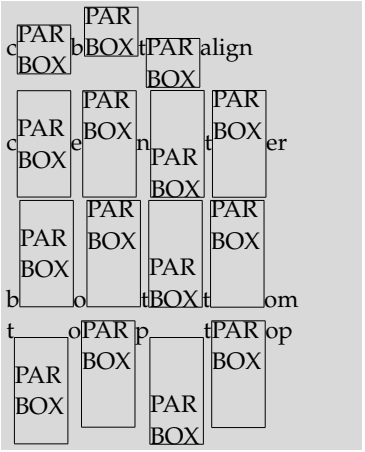
인자 수에 따른 매크로 대치는 다음과 같다. 여기에서 \relax는 아무런 일도 하지 않는 원시명령이다.

```
\parbox{#1}{#2}                 ⇒  \@iiiparbox{c}{\relax}[s]{#1}{#2}
\parbox[#1]{#2}{#3}             ⇒  \@iiiparbox{#1}{\relax}[s]{#2}{#3}
\parbox[#1][#2]{#3}{#4}         ⇒  \@iiiparbox{#1}{#2}[#1]{#3}{#4}
\parbox[#1][#2][#3]{#4}{#5}     ⇒  \@iiiparbox{#1}{#2}[#3]{#4}{#5}
```

이제 \@iiiparbox 매크로가 어떤 역할을 하는지 살펴보자. 아래 정의에서 이 매크로도 \leavevmode로 시작하는 것을 기억하자.

21. 이미 각주 11에서 옵션의 의미를 보았다. 단지 \hss를 \vss로 바꾸고 \unhbox를 \unvbox로 바꾸어 이해하면 된다. 또한 \parbox에서 주어진 옵션 [t]는 [l]과 같고, [b]는 [r]과 같다.

```
\newcommand\X{PAR\par BOX}
c\fbox{\parbox[c]{7mm}{\X}}b%
 \fbox{\parbox[b]{7mm}{\X}}t%
 \fbox{\parbox[t]{7mm}{\X}}align\par
c\fbox{\parbox[c][14mm][c]{7mm}{\X}}e%
 \fbox{\parbox[c][14mm][t]{7mm}{\X}}n%
 \fbox{\parbox[c][14mm][b]{7mm}{\X}}t%
 \fbox{\parbox[c][14mm][s]{7mm}{\X}}er\par
b\fbox{\parbox[b][14mm][c]{7mm}{\X}}o%
 \fbox{\parbox[b][14mm][t]{7mm}{\X}}t%
 \fbox{\parbox[b][14mm][b]{7mm}{\X}}t%
 \fbox{\parbox[b][14mm][s]{7mm}{\X}}om\par
t\fbox{\parbox[t][14mm][c]{7mm}{\X}}o%
 \fbox{\parbox[t][14mm][t]{7mm}{\X}}p%
 \fbox{\parbox[t][14mm][b]{7mm}{\X}}t%
 \fbox{\parbox[t][14mm][s]{7mm}{\X}}op
```



예제 17. 옵션에 따른 \parbox의 변화

```
\long\def\@iiiparbox#1#2[#3]#4#5{%
  \leavevmode
  \@pboxswfalse
  \setlength\@tempdima{#4}%
  \@begin@tempboxa\vbox{\hsize\@tempdima\@parboxrestore#5\@@par}%
    \ifx\relax#2\else
      \setlength\@tempdimb{#2}%
      \edef\@parboxto{to\the\@tempdimb}%
    \fi
    \if#1b\vbox
    \else\if #1t\vtop
    \else\ifmmode\vcenter
    \else\@pboxswtrue $\vcenter
    \fi\fi\fi
    \@parboxto{\let\hss\vss\let\unhbox\unvbox
       \csname bm@#3\endcsname}%
    \if@pboxsw \m@th$\fi
  \@end@tempboxa}
```

박스의 가로 폭을 주는 네 번째 인자 #4는 4715째 줄에서 \@tempdima 레지스터에 저장된다. 4716째 줄의 \@begin@tempboxa는 박스의 크기 정보를 저장하는 역할을 하는 것을 이미 여러 번 보았다. 이때 \vbox로 만들어진 박스의 크기를 측정하는데 가로 폭 \hsize는 \@tempdima에 저장된 값으로 설정된다. \@parboxrestore 매크로는 문단 조성에 관한 파라미터들을 원래대로 돌리는 역할을 한다 (latex.ltx 4744째 줄). 마지막에 나오는 \@@par는 본래의 원시명령 \par를 의미한다.[22] 저장된 크기 정보는 박스의 높이를 설정하는 #2 인자에서만 사용할 수 있다는 점에 유의하자.

22. \let\@@par=\par (latex.ltx 558째 줄) 원시명령 \par는 다른 패키지 또는 사용자가 내용을 바꿀 수 있기 때문에 LaTeX은 본래의 의미를 \@@par에 먼저 저장한 후 내부에서 \par 대신 \@@par를 사용한다.

박스를 만드는 과정은 4717째 줄에서 시작한다. 먼저 두 번째 인자 #2가 주어지면 이 값을 \@tempdimb에 저장한 후 \@parboxto를 설정한다. 즉, 두 번째 인자의 유무에 따라 나중에 4726째 줄에서 \vbox의 확장형을 사용할 지 결정한다. 4721–4725째 줄에서는 첫 번째 인자 [b|t|c]에 따라 서로 다른 원시명령 \vbox, \vtop, 그리고 \vcenter를 선택한다.[23] 마지막으로 4726째 줄에서 #3에서 주어진 세로 정렬방법에 따라 박스 레지스터 \@tempboxa에 저장해 두었던 박스의 내용을 부린다.

예제 16의 두 번째 결과를 \parbox를 이용해 만들려면 박스의 가로 폭을 어떻게 주느냐에 대한 어려움이 있다. LaTeX 사용자들이 이 매크로에 대한 매력을 느끼지 못하는 큰 이유 중의 하나가 여기에 있다. 예제 18은 이 문제를 \settowidth 매크로를 이용해 해결하는 방법을 제시한다.

```
98  \newlength\mylen
99  \settowidth\mylen{C}%
100 A\parbox[b]{\mylen}{B\par C}D%
101  \settowidth\mylen{E}%
102  \parbox[t]{\mylen}{E\par F}G
```

```
 B
ACDEG
   F
```

예제 18. \settowidth를 이용한 \parbox의 가로 폭 설정

박스 레지스터를 다루는 매크로들이 있는 것처럼 글루 레지스터를 다루는 매크로들도 있다. 예제 18의 98째 줄은 LaTeX에서 \newlength 매크로를 이용해 새로운 글루 레지스터를 만드는 방법을 보여준다. 이렇게 만든 레지스터에는 길이를 넣을 수 있는데 직접 입력하는 것은 물론, 99째 줄처럼 \settowidth, \settoheight, \settodepth 매크로를 이용해 주어진 박스의 폭, 높이, 그리고 깊이를 추출할 수 있다 (latex.ltx 1878–1880째 줄 참조).

6.2 문단 박스를 만드는 LaTeX 환경 minipage

문단 박스를 만들 때 \parbox 매크로에 비해 minipage 환경을 많이 사용하는 경향이 있다. 매크로 \parbox와 달리 minipage 환경 내부에서 다른 환경을 쓸 수 있을 뿐더러 각주도 달 수 있기 때문이다 (예제 19 참조). 즉, minipage 환경은 이름 그대로 작은 판면의 역할을 한다.

```
\begin{minipage}[정렬점위치][박스높이][내부정렬]{박스폭} 내용 \end{minipage}
```

이 환경은 박스의 폭을 지정하는 인자를 제외하고 세 개의 옵션을 받아들인다. 내용이 인자로 들어가는 \parbox를 환경 매크로로 확장한 것이라 볼 수 있으며 옵션의 사용법도 \parbox와 완전히 동일하다. 이 환경을 이용한 유용한 예제들은 [5, pp. 863–865]에서 찾을 수 있다.

23. 원시명령 \vcenter는 \vbox나 \vtop과 달리 수식모드에서만 사용할 수 있다. 코드에서 \if@pboxsw 스위치를 사용한 이유가 여기에 있다.

```
\begin{minipage}[b]{.5\textwidth}
First!\footnote{1st footnote}
\end{minipage}%
\begin{minipage}[t]{.5\textwidth}
\addtocounter{mpfootnote}{1}
Second!\footnote{2nd footnote}
\end{minipage}
```

First![a] Second![b]

a. 1st footnote

b. 2nd footnote

예제 19. 각주도 달 수 있는 minipage 환경

7 맺는 말

TEX이 문서에 나열된 글자와 그림들을 조판하는 방식은 금속활자를 이용한 근대적 활판 인쇄술의 그것과 상당히 유사하다. 금속활자를 가로로 나열해 하나의 줄을 만드는 것은 TEX에서 글자들의 박스를 이용해 수평모드에서 하나의 줄 박스를 만드는 것으로 이해할 수 있다. 그리고, 여러 개의 줄들을 하나의 판면 위에 세로로 조성하는 것은 수직모드에서 줄 박스들을 나열해 하나의 판면 박스를 만드는 것과 같다. 그러므로 박스와 글루는 TEX 조판 방식을 이해하는 열쇠가 된다.

여러가지 TEX 박스를 만드는 원시명령들을 제대로 이해하는 것은 그리 간단한 일이 아니다. 원시명령들의 조합으로 만들어진 LATEX 매크로는 이러한 원시명령들에 비해 이해하기 쉬울 뿐만 아니라 사용하기 편리하다. 이 글에서는 TEX 박스를 만드는 LATEX 매크로들이 어떠한 원리로 TEX 박스를 만드는지, 그리고 어떠한 방법으로 이러한 매크로들을 활용할 수 있는지 몇 가지 예제를 통해 살펴보았다. LATEX 사용자들이 보다 다양한 방법으로 문서를 만들 수 있고, 나아가 자신의 매크로 패키지를 만드는 데 조금이나마 도움이 되길 바란다.

참고 문헌

1. 조진환, TEX과 타이포그래피에 관한 소고,『韓國數學教育學會誌 시리즈 E』19권 4호 (2005), 823–837.
2. Victor Eijkhout, *TEX by Topic, A TEXnician's Reference*, Addison-Wesley, 1992. http://www.cs.utk.edu/~eijkhout/texbytopic-a4.pdf
3. Donald E. Knuth, *The TEX book*, Addison-Wesley, 1986.
4. Helmut Kopka and Patrick W. Daly, *Guide to LATEX*, 4th ed., Addison-Wesley, 2004.
5. Frank Mittelbach and Michel Goossens, *The LATEX Companion*, 2nd ed., Addison-Wesley, 2004.