

현대적 L^AT_EX 활용 II

NOVA DE HI

KOREAN T_EX USERS GROUP

21 Nov, 2021



INTRODUCTION

- 공주대학교 문서작성 워크숍 발표자료
(http://wiki.ktug.org/wiki/wiki.php/LaTeXWorkshop/2014?action=download&value=modernlatex_3.pdf)

‘현대적 L^AT_EX 활용’(2014)

- 공주대학교 문서작성 워크숍 발표자료
(http://wiki.ktug.org/wiki/wiki.php/LaTeXWorkshop/2014?action=download&value=modernlatex_3.pdf)
- 당시의 발표 내용을 수정하고 보완하려 한다.

- L^AT_EX 2_ε (1994). 거의 30년 정도가 되어간다.
- L^AT_EX 3는 처음에는 새로운 버전을 준비하는 프로젝트였다가 후일 L^AT_EX 2_ε의 관리 향상 프로젝트로 인식되기 시작하였다.
- L^AT_EX 3의 핵심 “L^AT_EX Programming Layer”인 EXPL3가 2020년부터 L^AT_EX kernel format에 완전히 포함되었고, ε-T_EX 엔진이 확장되어, 이전과는 다른 L^AT_EX이 되었다고 해도 좋다. 이제 많은 패키지들이 xparse와 expl3에 의존하여 작성되고 있다.
- 그러나, 이전의 L^AT_EX 언어는 그대로 활용할 수 있다. 새로운 것은 이전 포맷에서 실행되지 않지만 이전의 것은 (거의) 그대로 실행된다.

EXPL3를 알아야 할까?

- 캐주얼한 일반 사용자라면 굳이 몰라도 되지 않을까?
- 조금 심도 있게 사용하려면, `xparse`는 (되도록) 필수.
- 뭘 좀 더 해보려면, `expl3`은 아는 것이 좋다.

XPARSE

- texdoc xparse
- 2016 문서작성 워크숍: 조인성, “xparse: high-level document command parser,” <http://wiki.ktug.org/wiki/wiki.php/LaTeXWorkshop/2016?action=download&value=kts2016-xparse-updated.pdf>
- 2020 문서작성 워크숍: 조인성, “xparse: 효율적인 tikz 코딩,” <http://wiki.ktug.org/wiki/wiki.php/LaTeXWorkshop/2020?action=download&value=tikz-xparse.pdf>

간단한 연습

두 개의 인자를 받아들이는데 첫번째 인자는 **default**가 있는 경우의 명령을 정의해보자. (이 목적을 위한 **twoopt** 패키지가 있으나 여기서는 문제삼지 않는다.)

전통적인 정의 방법


```
\makeatletter
\newcommand\mycommand{\@ifnextchar [%
    {\mycommand@i}{\mycommand@i [<default-for-1>]}}%
}
\def\mycommand@i [#1]{\@ifnextchar [%
    {\mycommand@ii{#1}}{\mycommand@ii{#1} [<default-for-2>]}}
}
\def\mycommand@ii#1 [#2] #3{%
    % Do stuff
}
\makeatother
```


간단한 연습(계속)

xparse 방법

```
\NewDocumentCommand \mycommand{0{default}om}  
{ #1, \IfNoValue{#2}{..}{...} #3 }
```

[연습] `\test` 명령을 작성하시오. 옵션 인자는 2개이고 첫번째 것은 배경색, 두번째 것은 문자전경색. 옵션인자가 없으면 배경색을 `gray!50`로 합니다.

예) `\test{안녕}` → 

`\test[red!50]{안녕}` → 

`\test[blue!30][yellow]{안녕}` → 

간단한 연습 (계속) XPARSE + EXPL3

```
\ExplSyntaxOn
\keys_define:nn { mytest }
{
    bgcolor    .tl_set:N = \l_optbgcolor_tl,
    txtcolor   .tl_set:N = \l_opttxtcolor_tl
}
\NewDocumentCommand \test { o m }
{
    \IfValueT { #1 }
    { \keys_set:nn { mytest } { #1 } }
    ...
}
\ExplSyntaxOff

\test [bgcolor=blue!30,txtcolor=yellow]{안녕}
```

III (TABULAR)

- 2014년 당시 각광받던 `tabu` 패키지를 사용하기 힘들어진 지금, 대안으로 `tabularray` 패키지를 추천한다.
- 표에 관련된 그 동안의 여러 패키지를 통한 해법을 이것으로 대신할 수 있다.
- 한 줄 요약: `tabu` 대신 `tabularray`

- 2014년 당시 각광받던 **tabu** 패키지를 사용하기 힘들어진 지금, 대안으로 **tabularray** 패키지를 추천한다.
- 표에 관련된 그 동안의 여러 패키지를 통한 해법을 이것으로 대신할 수 있다.
- 한 줄 요약: **tabu** 대신 **tabularray**
- Tabular 관련된 많은 고민을 모아 놓은 KTUG Faq ‘Tabular환경’. 이 고민이 대부분 말끔하게 해결된다.

0-lines.tex

- 가로줄만으로 이루어지는 표, 첫 줄과 마지막 줄만 조금 굵게 그어보자. (`booktabs`, `mdwtab`, etc.)
- 첫 행과 두번째 행 사이에 겹가로줄을 그려보자. (`hhline`)
- 가로줄과 세로줄 중의 일부를 점선으로 그려보자. (`arydshln`)
- 줄에 굵기와 색상을 주어보자.

1-cells.tex

- 셀의 상대 위치 정렬을 `q` 컬럼 타입과 `\SetCell`로 해결해보자. (`array`)
- 둘 이상의 행에 걸치는 셀을 작성해보자. (`multirow`)
- 둘 이상의 열에 걸치는 셀을 작성해보자. (`\multicolumn`)
- `tabu`의 `x` 컬럼 타입은 완전히 지원된다. (`tabu`, `tabularx`)
- 셀 안의 행바꿈을 별도의 박스를 사용하지 않아도 된다. (`makecell`)
- 행이나 열, 셀에 색상을 줄 수 있다. (`colortbl`)

2-mores.tex

- `\NewColumnType` (`tabularx`, `tabulary`, `array`, `tabu`)
- 수식(`math mode`) 안의 표. 표 안의 수식.
- 긴 표(`long table`)

2-mores.tex

- `\NewColumnType` (`tabularx`, `tabulary`, `array`, `tabu`)
- 수식(math mode) 안의 표. 표 안의 수식.
- 긴 표(long table)

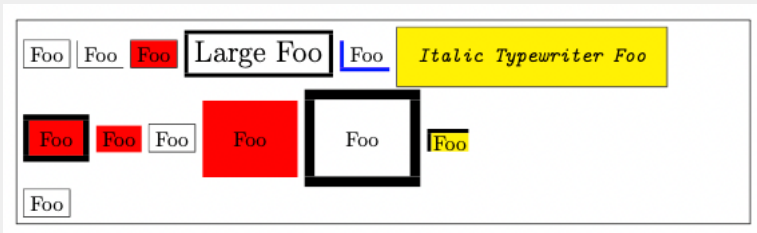
3-progress.tex

* 옛날 KTUGCollection 기본 문서로 제공하던 Progress님의 `first.tex`에 나오는 표 예제를 `tabularray`로 구현해본다.

TCOLORBOX

- 2014년에 mdfamed 추천한 것을 취소하고(...) tcolorbox의 사용을 권함.
- 발표자가 다른 기회에 작성한 박스 두 개, 그리고 ischo님의 example 환경을 하나 소개함. (tcbtest.tex)
 - ▶ theorem-류 박스 샘플
 - ▶ 시험문제지용 “보기” 박스
 - ▶ listing과 그 결과를 디스플레이하는 box

- \fbox를 대체하는 \efbox를 정의.
- key-value 방식의 옵션으로 다양한 fbox를 그릴 수 있다.



TOC

- memoir가 강력한 toc 관련 명령 세트를 제공하나, 배우기가 어렵다.
- 즉흥적 toc, toc 스타일링, 부분 toc를 위하여 etoc 패키지가 있음.
- 단, etoc는 memoir/oblivoir와는 함께 쓰지 못한다.
- oblivoir의 부속 패키지인 obchaptertoc 패키지는 oblivoir 문서의 ‘장 표제’를 만드는 데 주로 쓰이는 것.

- memoir가 강력한 toc 관련 명령 세트를 제공하나, 배우기가 어렵다.
- 즉흥적 toc, toc 스타일링, 부분 toc를 위하여 etoc 패키지가 있음.
- 단, etoc는 memoir/oblivoir와는 함께 쓰지 못한다.
- oblivoir의 부속 패키지인 obchaptertoc 패키지는 oblivoir 문서의 ‘장 표제’를 만드는 데 주로 쓰이는 것.
- 예제: obchaptertoctest.tex

FLOATS

- float 조판을 위한 key-value 인터페이스 제공
- fig, tab, figure, table, figbox, parbox, arbitrary float, subfigs, subfloats, wraps, marginfigure, margintable
- captionof 또는 fixedcaption을 간단히 처리할 수 있음



Additional text. Text text text text text.

More paragraphs.

Figure 19: A \keyfig[M]

Example 22: Using \keyfig[M]

Code:

```
\keyfig[M]{c={A \cs(keyfig)\optn{[M]}},l=fig:keyfigm,ft,  
t=Additional text.  
Text text text text text text.
```

More paragraphs.

```
}{image2}
```

Result:

Figure 19

INDICES

- `makeindex`, `xindy` 외부 명령 실행 자동화 (with `-shell-escape`)
- `theindex` 환경 설정을 위한 `key-value` 옵션 제공
- 기타 많은 편의 기능을 제공한다. 새로울 것은 없지만 사용성을 높임.
- `splitindex`와 같이 외부 유틸리티를 실행해야 하는 경우와 호환.
(`multind`는 이미 `imakeidx`로 가능하므로 별도로 로드하지 못하게 하고 있다.)

재미난 거

- 수식을 ‘조판’ 하고 임의의 값에 대하여 ‘계산’을 행한다.
- $48 \div 2(9 + 3) = 288$

```
\eval{\[
  \left(
    \frac{1-4\sin^2\frac{m}{3n}\pi}
    {2\sin^2\frac{m}{3n}\pi}
  \right)
  \sin\frac{2m-3}{3n}\pi\sin\frac{m-3}{3n}\pi
\]}[m=2,n=5]
```

$$\left(\frac{1 - 4 \sin^2 \frac{m}{3n} \pi}{2 \sin^2 \frac{m}{3n} \pi} \right) \sin \frac{2m-3}{3n} \pi \sin \frac{m-3}{3n} \pi = -0.044193, \quad (m = 2, n = 5)$$

LONGDIVISION

```
\longdivision{100}{22}  
\quad  
\intlongdivision{100}{22}
```

$$\begin{array}{r} 4.5\overline{4} \\ 22 \overline{)100.00} \\ \underline{88} \\ 12.0 \\ \underline{11.0} \\ 1.00 \\ \underline{88} \\ 12 \end{array}$$

xlop 패키지에 의한 프랑스식 나눗셈과 비교하여 보자.

$$\begin{array}{r|l} 100 & 3 \\ 10 & 33.3 \\ 10 & \\ 1 & \end{array}$$

source
`\opdiv[period]{100}{3}`

- 다항식 계산 패키지.
- 조립제법(Horner's scheme), 다항식 덧셈 뺄셈 곱셈 나눗셈, 인수분해

$$\begin{array}{r}
 X^2 + 2X + 2 \\
 X - 1 \overline{) X^3 + X^2 - 1} \\
 \underline{-X^3 + X^2} \\
 2X^2 \\
 \underline{-2X^2 + 2X} \\
 2X - 1 \\
 \underline{-2X + 2} \\
 1
 \end{array}$$

$$1 \left| \begin{array}{cccc}
 1 & 1 & 0 & -1 \\
 & 1 & 2 & 2 \\
 \hline
 1 & 2 & 2 & 1
 \end{array} \right.$$

```
\polylongdiv{X^3+X^2-1}{X-1}
```

```
\polyhornerscheme[x=1]{x^3+x^2-1}
```

```
\polyfactorize {(X-1)(X-1)(X^2+1)} (X^2 + 1) (X - 1)^2
```


- ‘계산’을 해주는 것은 `numerica`와 같지만 인터페이스가 다르다.
- 그 대신 이런 것도 가능.

Ex. 49

$$\begin{bmatrix} 1 & -1 \\ 3 & 0 \end{bmatrix}^T = \begin{bmatrix} 1 & 3 \\ -1 & 0 \end{bmatrix}$$

```
\TRANPOSEMATRIX(1,-1;3,0)%
(\sola,\solb;\solc,\sold)
$\begin{bmatrix}
  1 & -1 \\ 3 & 0
\end{bmatrix}^T=\begin{bmatrix}
  \sola & \solb \\ \solc & \sold
\end{bmatrix}$
```

For the $f(t) = (t^2, t^3)$ function we have

$$f(4) = (16, 64), f'(4) = (8, 48)$$

- 벡터 미분(!)도 한다고 한다.

이 section에서 소개한 몇 가지 ‘계산’ 패키지들은, 그 유용성 때문에 소개한 것은 아니다. 실제로 복잡한 계산을 더 잘 해주는 프로그램이 많으므로 계산은 계산 전문에게 맡기는 것이 좋다.

다만, 재미삼아, 그리고 간단한 계산 결과가 필요할 때에 이런 패키지를 사용할 수 있을 것이다.

또 계산은 `expl3`의 `l3fp`와 `l3int`로도 해낼 수 있을 것이다. 다만 코딩은 자신의 몫이겠지만.

T_EX SYSTEM INSTALLATION

- Windows는 2021년 현재 아직까지 win32가 기본이다. (win64로 조만간 이행할 듯함)
- 그 대신, Windows 10/11이 WSL이라는 강력한 도구를 제공함으로써, Linux의 바이너리를 (GUI 프로그램까지 포함하여) 쉽게 실행하게 되었다.

- Windows는 2021년 현재 아직까지 win32가 기본이다. (win64로 조만간 이행할 듯함)
- 그 대신, Windows 10/11이 WSL이라는 강력한 도구를 제공함으로써, Linux의 바이너리를 (GUI 프로그램까지 포함하여) 쉽게 실행하게 되었다.
- Windows용 MiK_TE_X은 여전히 Windows에서 주류인 듯. MiK_TE_X은 64bit 바이너리를 가지고 있고, 특유의 MPM 시스템이 원활하게 동작하며, 매년 새 버전을 설치하지 않아도 된다. MiK_TE_X-Console 유틸리티를 통하여 항상 최신 버전 유지 가능.

- Windows는 2021년 현재 아직까지 win32가 기본이다. (win64로 조만간 이행할 듯함)
- 그 대신, Windows 10/11이 WSL이라는 강력한 도구를 제공함으로써, Linux의 바이너리를 (GUI 프로그램까지 포함하여) 쉽게 실행하게 되었다.
- Windows용 MiK_TE_X은 여전히 Windows에서 주류인 듯. MiK_TE_X은 64bit 바이너리를 가지고 있고, 특유의 MPM 시스템이 원활하게 동작하며, 매년 새 버전을 설치하지 않아도 된다. MiK_TE_X-Console 유틸리티를 통하여 항상 최신 버전 유지 가능.
- Mac OS는 Mac_TE_X. Linux는 각자 알아서.

간략한 T_EX LIVE?

- **tectonic**은 바이너리 하나만 설치하면 끝이다. X_YL^AT_EX 위주로 라텍스를 사용하는 사람이고, 최신판 패키지가 꼭 필요한 게 아니라면(현재 TL2020 베이스) 고려해 봄직하다.
- **TinyT_EX**은 R 사용자 측에서 시작된 ‘작은 규모의 T_EX 시스템’. 그러나 일단 설치하고 나면 사실상 T_EX Live처럼 쓸 수 있기 때문에 이를 선호할 수 있다.
- **Docker** 개발 환경이 유행하면서 T_EX Live 또는 MiK_TE_X Docker 이미지들이 많이 제작되었다. KTUG에 소개한 http://www.ktug.org/xe/index.php?mid=KTUG_open_board&document_srl=255839도 일별 요망. 이 경우도 Docker 실행 환경이 갖추어져 있다면 간단히 `pull` 명령 하나로 즉시 T_EX을 이용할 수 있다.

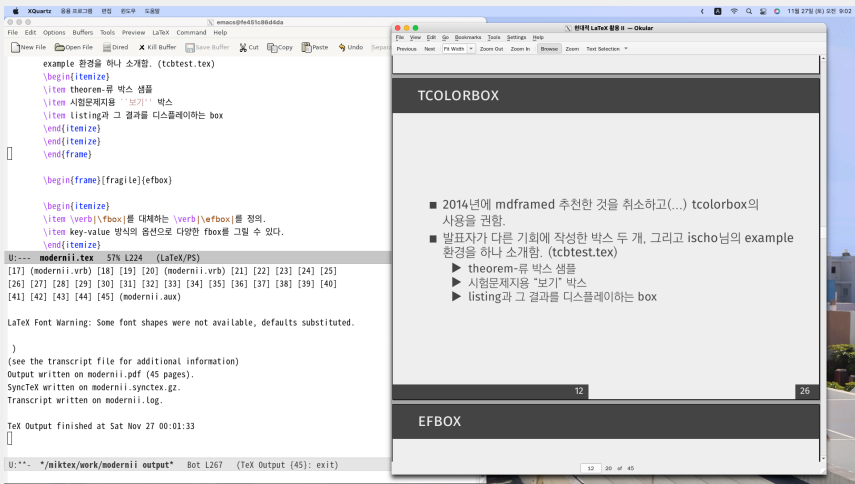
- 좋은 에디터를 손에 익히는 것은 생산성 향상의 근본이다.
- TEX works는 뭔가 간소하고, Mac의 TEX Shop 같은 만족감을 주지 못한다.
- 유명하다는 몇몇 에디터도 어딘가 불만족스러울 수 있음.

- 좋은 에디터를 손에 익히는 것은 생산성 향상의 근본이다.
- $\text{T}_{\text{E}}\text{X}$ works는 뭔가 간소하고, Mac의 $\text{T}_{\text{E}}\text{X}$ Shop 같은 만족감을 주지 못한다.
- 유명하다는 몇몇 에디터도 어딘가 불만족스러울 수 있음.
- Emacs, Vim이 손에 익었다면 적극적으로 고려할 것.
- Vim에 대해서는 2020년 KTS에서의 이재호의 발표, <https://github.com/Zeta611/texnical-vim-kts-conf-2020> 참조.

- 좋은 에디터를 손에 익히는 것은 생산성 향상의 근본이다.
- $\text{T}_{\text{E}}\text{X}$ works는 뭔가 간소하고, Mac의 $\text{T}_{\text{E}}\text{X}$ Shop 같은 만족감을 주지 못한다.
- 유명하다는 몇몇 에디터도 어딘가 불만족스러울 수 있음.
- Emacs, Vim이 손에 익었다면 적극적으로 고려할 것.
- Vim에 대해서는 2020년 KTS에서의 이재호의 발표, <https://github.com/Zeta611/texnical-vim-kts-conf-2020> 참조.
- VS Code는 좀 무겁고 사용환경 설정이 번거롭지만 적어도 Windows에서 이만한 에디터가 없다는 생각임.

- 좋은 에디터를 손에 익히는 것은 생산성 향상의 근본이다.
- T_EXworks는 뭔가 간소하고, Mac의 T_EXShop 같은 만족감을 주지 못한다.
- 유명하다는 몇몇 에디터도 어딘가 불만족스러울 수 있음.
- Emacs, Vim이 손에 익었다면 적극적으로 고려할 것.
- Vim에 대해서는 2020년 KTS에서의 이재호의 발표, <https://github.com/Zeta611/technical-vim-kts-conf-2020> 참조.
- VS Code는 좀 무겁고 사용환경 설정이 번거롭지만 적어도 Windows에서 이만한 에디터가 없다는 생각임.
- 좋은 에디터 좀 소개시켜 주세요!

MIKTeX + EMACS DOCKER



- $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 패키지군은 종류도 많고 용량도 크지만 다양한 문제에 대하여 대부분 해결책이 제시되어 있다.
- 수많은 패키지를 다 알지 않더라도, 자신의 목적에 맞는 명령이나 함수를 제작하여 사용하는 것도 좋다.
- 문제가 생기면 KTUG에 질문한다. 답변을 얻었으면 감사한다. 문제를 스스로 해결했으면 KTUG에 자랑한다. 즐거운 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 생활.

감사합니다

Happy L^AT_EX'ing