

The **alttex** package

Arno L. Trautmann*

Version 0.a, 2008 December 17

This is the package **alttex** which will try to give an experimental new way to write \LaTeX code. So far it is mostly done with very dirty code and actually it's a collection of things that come into my mind during boring lectures. Maybe someone will have fun with the following code fragments.

Contents

*arno.trautmann@gmx.de

1 introduction

The problem I have with L^AT_EX¹ is the antique way of typing. Because most people still use a hopelessly outdated keyboard layout („qwerty“ or slightly adapted versions of that), L^AT_EX doesn't make use of some cool features. I'm not talking about writing chinese or arabic text! Maybe this example will make the idea clear:

In standard L^AT_EX, one has to write

This is the normal text, then comes the itemization:

```
\begin{itemize}
  \item text for first item
  \item \begin{itemize}
    \item this is an item inside an item...
    \item[$\rightarrow$] Here an item with a formula: $\int_a^b x^2 dx$
  \end{itemize}
  \item and the outer itemize goes on...
\end{itemize}
```

Using this package and having a superior keyboard layout², you can simply write:³

This is the normal text, then comes the itemization:

- text for first item
- - this is an item inside an item
 - [⇒] Here an item with a formula: $\int_a^b x^2 dx$
- and the outer itemize goes on...

And your normal text goes on...

Well, actually I'm lying now because this is not fully implemented so far. But it's the aim of this package to provide this – besides many, many other funny and cool things. The aim is to offer a more „wysiwyg“ way, without loosing anything of logical markup. One still can re\define the • if he doesn't like the way his items look. I have just started to write the package, there will be much more stuff here in the future.

Ok, enough blahblah, now comes the code. We begin with the mostly uninteresting preamble stuff:

```
1 \ProvidesPackage{alttex}
```

¹I'll write L^AT_EX instead of X_YL^AT_EX—saves me two keystrokes. Most of the code below *only* works with X_YL^AT_EX. If you need support for [utf8]inputenc or LuaL^AT_EX, please contact the author.

²E.g. the ergonomic layout NEO.

³The lmodern font I'm using here does not have the symbol for the inner item , so we change to DejaVu Sans Mono here.

```

2
3 \RequirePackage{amsmath}
4 \RequirePackage{exscale}
5 \RequirePackage{ifxetex}
6 \ifxetex
7 \typeout{Loading XeTeX, everything's fine.}
8 \else
9 \typeout{^^J%
10 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!^^J%
11 ! This package can only be compiled with XeLaTeX.^^J%
12 ! pdfLaTeX cannot handle unicode the way it is used here.^^J%
13 ! If you want to have support for [utf8]inputenc, please contact the author.^^J%
14 ! If you want to use LuaLaTeX, give it a try:^^J%
15 ! comment out the lines 32,33,35-43.^^J%
16 ! Please e-mail me the result of your experiences!^^J%
17 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!^^J%
18 }
19 \errmessage{No XeLaTeX, no alttx. See the log for more information.}
20 \endinput
21 \fi
22

```

We need `exscale` to write really big formulae, and `ifxetex` to check whether one uses the correct engine.

2 Textmode

2.1 no escape

`\noescape` You want to write plain text. Maybe you're annoyed by always escaping characters like `_` `#` `&` `{` `}` `$` `~` and so on. `\noescape` allows you to never escape anything—except the `\`, which still might be used for `\textit{}` or so. Or maybe not... because the `{` `}` are not escaped. Have to think about this one. Maybe the `\` will be redefined to define `{` `}` by itself.

```

23 \def\noescape{
24 \catcode\_= 11%
25 \catcode\^= 11%
26 \catcode\#= 11%
27 \catcode\&= 11%
28 %\catcode\{= 11%
29 %\catcode\}= 11%
30 \catcode\$= 11%
31 \catcode\~= 11%
32 \makeatletter% I just noticed this is not necessary... but I'll leave it for some strange \thin@gs or
33 \catcode\%= 11
34 }
1

```

`\oldescape` Of course this has to be reset when doing anything like formula, tabular etc. Maybe I will be able to change the behaviour automatically. This idea has been inspired by a discussion on the ConTeXt mailinglist.

```

35 \def\oldescape{
36   \catcode`\%= 14%
37   \catcode`\_ = 8%
38   \catcode`\^ = 7%
39   \catcode`\# = 6%
40   \catcode`\& = 4%
41   \catcode`\{ = 1%
42   \catcode`\} = 2%
43   \catcode`\$ = 3%
44   \catcode`\~ = 13%
45   \makeatother%
46 }

```

3 Math stuff

3.1 braces

`\newbraces` Now this is something most L^AT_EX-beginners don't recognize and wonder why the
`\oldbraces` formula looks so ugly: The braces () do not fit to the height of the formula. This can be achieved by putting `\left` and `\right` in front of the braces. But actually, this is annoying! In almost any case you want this behaviour, so this should be the standard. So we redefine the way braces are handled. With `\newbraces` the () always fit. If you prefer the normal L^AT_EX way, use `\oldbraces` to reset everything. This new behaviour should be extended to other characters like | [{ < and so on. Maybe in version 0.0.1...

I would have never been able to implement this without the help of the mailinglist members of `tex-d-l@listserv.dfn.de`!

The redefinition of `\mathstrut` is necessary when using `amsmath` (you will use `amsmath` when typesetting formulae, won't you?), because the height of formulae is determined by the height of a brace. But using () as `\active` characters, we need another brace here. So we take [. This will probably also change. But the code is working fine for ().

```

47 \makeatletter
48 \def\resetMathstrut@{%
49   \setbox\z@\hbox{%
50     \mathchardef\@tempa\mathcode`\[ \relax
51     \def\@tempb##1"##2##3{\the\textfont"##3\char"}%
52     \expandafter\@tempb\meaning\@tempa \relax
53   }%
54   \ht\Mathstrutbox@ \ht\z@ \dp\Mathstrutbox@ \dp\z@
55 }
56 \makeatother
57
58 \edef\oldbraces{

```

Maybe one could "temporarily hardcode" the height of [and then use this...

```

59 \mathcode`(\the\mathcode`(
60 \mathcode`) \the\mathcode`)
61 }
62 \begingroup
63 \catcode`\active \xdef({\left\string(}
64 \catcode`\active \xdef){\right\string)}
65 \endgroup
66 \def\newbraces{
67
68 \mathcode`("8000
69 \mathcode`) "8000
70 }

```

`hugedisplaymath` Sometimes, especially in presentations, you might need an really big formula. Imagine two hours of struggle with transformations—and finally there is the beautiful formula. Now you can say

```
\begin{hugedisplaymath} E = mc^2 \end{hugedisplaymath}
```

There should be several steps of size, maybe.

```

71 \def\hugedisplaymath{
72 \makeatletter
73 \makeatother
74 \Huge
75 \begin{equation*}
76 }
77 \def\endhugedisplaymath{
78 \end{equation*}
79 }

```

4 itemize and similar things

4.1 itemize with a single character

- instead of `\item` Here we use an active character (mostly a unicode character bullet •) for the whole construct. And another one for nested itemizations (like a triangular bullet ►)

This does not—guess it— work correctly so far. I’m trying to find a tricky way so that the ending character is not necessary any more. So far one has to end an itemize with something like an -. There will also be a possibility to change the characters responsible for the whole action.

```

80
81 \catcode`\.=\active
82 \catcode`\ =\active
83 \makeatletter
84 \def\notinside{}
85 \def\inside{}
86 \let\insideitemizei\outside

```

```

87 \let\insideitemizeii\outside
88
89 \def•{
90   \ifx\insideitemizei\inside
91     \item
92   \else
93     \begin{itemize}
94       \global\let\insideitemizei\inside
95       \item
96     \fi
97 }
98
99 \def {
100   \ifx\insideitemizeii\inside
101     \item
102   \else
103     \begin{itemize}
104       \global\let\insideitemizeii\inside
105       \item
106     \fi
107 }
108

```