# 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 X⅁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 LaTeX[1] is the antique way of typing. Because most people still use a hopelessly outdated keyboard layout („qwerty" or slightly adapted versions of that), LaTeX doesn't make use of some cool features.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 LaTeX, 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 layout[2], 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: $∫_a^b x² dx$

• 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}
```

---

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

[2] E. g. the ergonomic layut NEO.

[3] The 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 alttex. See the log for more information.}
20  \endinput
21 \fi
22
```

We need `exscale` to write really big formulae, and `ifxetex` to check wether 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 }
```

l

`\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. Thas 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 LaTeX-beginners don't recognize and wonder why the
`\oldbraces`  formula looks so ugly: The braces () do not fit to the hight 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 LaTeX 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 hight of formulae is determinated by the hight 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 ( ).

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

```
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{
```

```
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 – (em-dash). There will also be a possibility to change the characters responsible for the whole action.[4]

\newitemi
\newitemii The following ugly peace of code is writen by me, defining the conditional insertion of the \begin{itemize}. This will be assigned to an active character using \makeitemi and \makeitemii, respectively.

```
80 \def\notinside{}
81 \def\inside{}
```

---

[4]The triangular bullet sign does not appear here – the font is lacking it…

```
82 \let\insideitemizei\outside
83 \let\insideitemizeii\outside
84
85 \def\newitemi{
86   \ifx\insideitemizei\inside
87     \expandafter\item%
88   \else
89     \begin{itemize}
90       \global\let\insideitemizei\inside
91       \expandafter\item%
92   \fi
93 }
94
95 \def\newitemii{
96   \ifx\insideitemizeii\inside
97     \expandafter\item%
98   \else
99     \begin{itemize}
100      \global\let\insideitemizeii\inside
101      \expandafter\item%
102  \fi
103 }
```

Ok, the following code is stolen from the shortvrb package, and I don't understand anything of it. But I keep on trying... nevertheless, it's working fine, as far as I can see.

<div style="float:left">\makeitemi<br>\makeitemii</div>

With this macro, you can define the character you want to use for first-level itemize. (Guess the sense of \makeitemii...) Default ist • for first-level and   for second-level. Maybe this will be extended till fourth level. More doesn't seem to make any sense.

```
104 %
105 \makeatletter
106 \def\makeitemi#1{%
107   \expandafter\ifx\csname cc\string#1\endcsname\relax
108     \add@special{#1}%
109     \expandafter
110     \xdef\csname cc\string#1\endcsname{\the\catcode`#1}%
111     \begingroup
112       \catcode`\~\active  \lccode`\~`#1%
113       \lowercase{%
114       \global\expandafter\let
115         \csname ac\string#1\endcsname~%
116       \expandafter\gdef\expandafter~\expandafter{\newitemi}}%
117     \endgroup
118     \global\catcode`#1\active
119   \else
120   \fi
121 }
122
```

```
123 \def\makeitemii#1{%
124   \expandafter\ifx\csname cc\string#1\endcsname\relax
125     \add@special{#1}%
126     \expandafter
127     \xdef\csname cc\string#1\endcsname{\the\catcode`#1}%
128     \begingroup
129       \catcode`\~\active  \lccode`\~`#1%
130       \lowercase{%
131       \global\expandafter\let
132           \csname ac\string#1\endcsname~%
133       \expandafter\gdef\expandafter~\expandafter{\newitemii}}%
134     \endgroup
135     \global\catcode`#1\active
136   \else
137   \fi
138 }
```

Now there are the two helperfunctions – no guess what they are really doing.

```
139 \def\add@special#1{%
140   \rem@special{#1}%
141   \expandafter\gdef\expandafter\dospecials\expandafter
142 {\dospecials \do #1}%
143   \expandafter\gdef\expandafter\@sanitize\expandafter
144 {\@sanitize \@makeother #1}}
145 \def\rem@special#1{%
146   \def\do##1{%
147     \ifnum`#1=`##1 \else \noexpand\do\noexpand##1\fi}%
148   \xdef\dospecials{\dospecials}%
149   \begingroup
150     \def\@makeother##1{%
151       \ifnum`#1=`##1 \else \noexpand\@makeother\noexpand##1\fi}%
152     \xdef\@sanitize{\@sanitize}%
153   \endgroup}
154 \makeatother
```

Finally, we set the default characters for itemi and itemii:

```
155 \makeitemi•
156 \makeitemii
```

And that's it. Happy altTEXing!