

An essential Guide to the dos and don'ts of \LaTeX 2 ϵ or obsolete Commands and Packages, and some more Mistakes to avoid

Original German Version
by Mark Trettin^{*}

English Translation
by Jürgen Fenn[†]

A revised and very slightly enlarged
Translation of Version 1.7
October 24, 2004

Abstract

This is the English version of “l2tabu”. Reading the German-language newsgroup on \TeX ¹ I found that quite a number of discussions are about obsolete or, say, ‘bad’ packages, and commands. So I decided to write this little overview to supply a practical guide to \LaTeX .

In this article I try to show the most common mistakes in using \LaTeX . I also give some hints on how to avoid them. This overview is neither meant to replace introductions such as `lshort` [8] nor the De- \TeX -FAQ [5, version 72] nor the UK FAQ [3, version 3.9]. My goal is just to give a small overview of how to write “good” \LaTeX 2 ϵ code.

I am grateful for any suggestions, improvements, or comments.

Copyright © 2004 by Mark Trettin and Jürgen Fenn.

This material may be distributed only subject to the terms and conditions set forth in the *Open Publication License*, v1.0 or later (the latest version is presently available at <http://www.opencontent.org/openpub/>).

^{*}email: Mark.Trettin@gmx.de

[†]email: juergen.fenn@gmx.de

¹de.comp.text.tex

Thanks to Ralf Angeli, Christoph Bier, Christian Faulhammer, Jürgen Fenn², Yvon Henel³, Yvonne Hoffmüller, David Kastrup, Markus Kohm, Thomas Lotze, Frank Mittelbach, Heiko Oberdiek, Walter Schmidt, Stefan Stoll, Emanuele Zannarini⁴ and Reinhard Zierke for tips, remarks, and corrections.

Barbara Beeton, Christoph Bier, Klas Elmgren, Yvon Henel, Hendrik Maryns, José Carlos Santos, Maarten Snee, Stefan Ulrich, Walter Schmidt, and Federico Zenith have contributed to the English version, making all sorts of suggestions or encouraging development.

If we have forgotten anyone who has contributed to this paper please send an email to the maintainer of the language version.

²Please send any comments on the English version directly to email: `juergen.fenn@gmx.de`

³French translation: `CTAN:info/l2tabu/french/l2tabufr.pdf`

⁴Italian translation: `CTAN:info/l2tabu/italian/l2tabuit.pdf`

Contents

1. “Deadly sins”	4
1.1. <i>a4.sty</i> , <i>a4wide.sty</i>	4
1.2. Modifying layout	4
1.3. Changing packages and document classes	4
1.4. Changing inter-line space using <code>\baselinestretch</code>	5
1.5. Paragraph indent and spread between paragraphs (<code>\parindent</code> , <code>\parskip</code>)	5
1.6. Separating maths formulae from continuous text using <code>\$\$...\$\$</code>	6
1.7. <code>\def</code> vs. <code>\newcommand</code>	6
1.8. Should I use <code>\sloppy</code> ?	6
2. Some obsolete commands and packages	7
2.1. Commands	7
2.1.1. Changing font style	7
2.1.2. Mathematical fractions (<code>\over</code> vs. <code>\frac</code>)	8
2.1.3. Centring text using <code>\centerline</code>	8
2.2. Class files and packages	9
2.2.1. <i>scrlettr.cls</i> vs. <i>scrlettr2.cls</i>	9
2.2.2. <i>epsf.sty</i> , <i>psfig.sty</i> , <i>epsfig.sty</i> vs. <i>graphics.sty</i> , <i>graphicx.sty</i>	9
2.2.3. <i>doubleSPACE.sty</i> vs. <i>setspace.sty</i>	9
2.2.4. <i>fancyheadings.sty</i> , <i>scrpage.sty</i> vs. <i>fancyhdr.sty</i> , <i>scrpage2.sty</i>	10
2.2.5. <i>isolatin.sty</i> , <i>umlaut.sty</i> vs. <i>inputenc.sty</i>	10
2.2.6. <i>t1enc.sty</i> vs. <i>fontenc.sty</i>	11
2.2.7. <i>natdin.bst</i> vs. <i>dinat.bst</i>	11
2.3. Fonts	11
2.3.1. <i>times.sty</i>	12
2.3.2. <i>mathptm.sty</i>	12
2.3.3. <i>pslatex.sty</i>	12
2.3.4. <i>palatino.sty</i>	12
2.3.5. <i>mathppl.sty</i>	13
2.3.6. Typesetting upright greek letters	13
2.3.7. <i>euler.sty</i>	13
3. Miscellaneous	13
3.1. Floats – “figure”, “table”	14
3.2. The appendix	14
3.3. Mathematical typesetting	14
3.4. Language-specific macros – <code>*name</code>	15
A. An example illustrating the effect of the <code>\sloppy</code> command	16

1. “Deadly sins”

In this section I probably have gathered together the most severe mistakes that appear again and again in `de.comp.text.tex`, leaving regulars either with a flush of anger or weeping with tears in their eyes. ; -)

1.1. *a4.sty*, *a4wide.sty*

These ‘two’ packages should no longer be used. You should delete them without a replacement from your \LaTeX source. Use class option `a4paper` instead. The layout produced by these or similar packages is more than questionable as far as typography is concerned. Even worse, there is more than one version of these packages around. Versions are incompatible with one another. They provide different settings for type area. You cannot even make sure that your document looks the same – or just as bad? – on someone else’s system as it does look when compiled on your own system.

1.2. Modifying layout

The margins produced by standard classes (*article.cls*, *report.cls*, *book.cls*) are often deemed too wide by Europeans using A4 paper. In this case you should use the respective classes (*scrartcl.cls*, *scrreprt.cls*, *scrbook.cls*) from the KOMA-Script bundle. You can also use the package *typearea.sty* which is part of KOMA-Script. The documentation included in the bundle provides some more information.

If you really need to use a type area altogether different from the one produced by *typearea.sty* please use the *geometry.sty*, or the *vmargin.sty* package. Do not try to modify layout using `\oddsidemargin` & friends.

Under no circumstances you should change `\hoffset`, or `\voffset`, unless you really understand what \TeX is doing here.

1.3. Changing packages and document classes

Never change \LaTeX classes (e.g., *article.cls*, *scrbook.cls*) or packages (style files, e.g., *varioref.sty*, *color.sty*) directly! If you do not want to make yourself a ‘container class’, or a `.sty` file of your own you should *copy* the class/style files, edit *the copy*, and save it as a *different* file using a *different* file name.

On how to make container classes see the De-TeX-FAQ [5, question 5.1.5].

Note: You should install any additional files or packages in the local texmf tree in your `$HOME` directory. Otherwise these changes will be overwritten when upgrading your \TeX distribution. Styles or packages you only need in one particular project or which you may want to hand on to someone you wish to share your project with may as well be saved in the current working directory. See the De-TeX-FAQ [5, question 5.1.4], or the UK FAQ [3, “Installing \LaTeX files“, section O, “Where to put new files“, question 125].

1. “Deadly sins”

1.4. Changing inter-line space using `\baselinestretch`

As a rule of thumb parameters always should be changed on the highest level possible within a user interface. So if you want to change inter-line space you can do so on three levels:

1. Either by using the *setspace.sty* package;
2. or by using the L^AT_EX command `\linespread{<factor>}`;
3. or by redefining `\baselinestretch`.

Redefining parameters such as `\baselinestretch` works on the lowest level available. This is why it should better be left to packages. The `\linespread` command is meant for this, so it is a better way to get more inter-line space. It is even better, though, to use *setspace.sty* which also takes care of space in footnotes and list environments that you normally don’t want to change when modifying inter-line space.

So if you just need some more spacing between lines, say, you would like to set spacing to one half or to double spacing, *setspace.sty* provides the easiest way to achieve this. However, if you only want to use fonts other than Computer Modern you may use `\linespread{<factor>}`. For instance, when using Palatino `\linespread{1.05}` would be appropriate.

1.5. Paragraph indent and spread between paragraphs (`\parindent`, `\parskip`)

It may make sense to change the indent of the first line in paragraphs (`\parindent`). However, please note the following:

- Never use absolute sizes (‘mm’, for instance) to change paragraph indent. Use sizes that depend on font size, such as ‘em’, e.g. The latter does *not* mean that indent adapts automatically to a change in font size. Rather, the value that goes with the font currently activated is used.
- Use L^AT_EX syntax for the job as this will usually be the least tricky way to do so. For instance this may make it easier to parse⁵ a L^AT_EX file through an external program, or script. It will be easier to maintain, too. This is to avoid compatibility problems with other packages as well (*calc.sty*).

Replace: `\parindent=1em` by `\setlength{\parindent}{1em}`

However in case you prefer some additional space between paragraphs to paragraph indent for marking the start of a new paragraph (‘zero paragraph indent’) do *not* use

```
\setlength{\parindent}{0pt}  
\setlength{\parskip}{\baselineskip}
```

⁵That is to say, analyse syntactically, or split up.

1. “Deadly sins”

The `\parskip` macro should be avoided as it will also modify list environments, table of contents etc., and headings.

The *parskip.sty* package, however, as well as the KOMA-Script classes go to some lengths to avoid these side effects. For how to use those KOMA-Script options (`parskip`, `halfparskip`, etc.) see scruien [4]. When using one of the KOMA-Script classes you do *not* need to load *parskip.sty*.

1.6. Separating maths formulae from continuous text using `$$. . . $$`

Please don’t do this! `$$. . . $$` is a Plain \TeX command. It will modify vertical spacing within formulae, rendering them inconsistent. This is why it should be avoided in \LaTeX (see section 3.3 on page 14; note the warning concerning `displaymath` along with the *amsmath.sty* package). What’s more, class option `fleqn` no longer works.

Replace: `$$. . . $$` by `\[. . . \]`

or

```
\begin{displaymath}
...
\end{displaymath}
```

1.7. `\def` vs. `\newcommand`

Always define your macros by way of `\newcommand{\<name>}{ . . . }`.

Never use `\def\<name>{ . . . }`. The main problem with `\def` is that no check is done on whether there already exists another macro of the same name. So a macro defined earlier may be overwritten without any error warning.

Macros may be redefined using `\renewcommand{\<name>}{ . . . }`.

He that knows *why* he needs to use `\def` will probably know of the commands’s pros and cons. He may well ignore this subsection.

1.8. Should I use `\sloppy`?

Frankly speaking, the `\sloppy` switch should not be used at all. Not even in the preamble of a document. For solving problems with line breaks in individual paragraphs in order to achieve quality typesetting you should

1. check whether the right hyphenation patterns, e.g., *(n)german.sty*, and T1 fonts have been loaded (see De-TeX-FAQ [5, Abschnitt 5.3]), or the UK FAQ [3, “Hyphenation”, section Q.7];
2. put your text in other words. You do not necessarily need to change the sentence the line break problem appears in. It may be enough to change the preceding, or the following sentence;

2. Some obsolete commands and packages

3. slightly change some parameters \TeX uses in breaking lines. Axel Reichert has suggested this solution⁶ on `de.comp.text.tex`:⁷

```
\tolerance 1414
\hbadness 1414
\emergencystretch 1.5em
\hfuzz 0.3pt
\widowpenalty=10000
\vfuzz \hfuzz
\raggedbottom
```

Note that warnings appearing with the above settings *really* should be taken into account. You *should* put your text in other words, then.

Only if these attempts should fail you may try to typeset the following paragraph more “loosely” using the `sloppypar` environment.

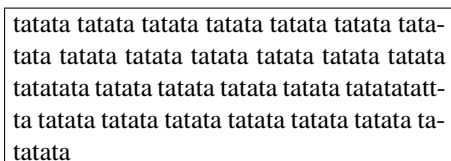


Figure 1: \LaTeX 's default settings

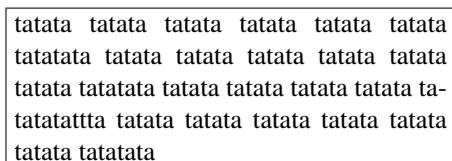


Figure 2: This demonstrates the effect of `\sloppy`

In figures 1 and 2 I have tried to show the effect of `\sloppy`. With “Times” the negative effects of `\sloppy` do not show as extremely as with, say, “Computer Modern”. The effect-in-principle, however, should become clear.

In `comp.text.tex` Markus Kohm has posted an example that shows this effect even better. With his kind permission I have put it in the appendix A on page 16.

2. Some obsolete commands and packages

Markus Kohm has written a Perl script you can test your files online with for the most common mistakes. See <http://kohm.de.tf/markus/texidate.html>. Please note, however, that this script is not a complete \TeX parser. This is why it will only check for the most common mistakes. Please test your file first, then post for help to a newsgroup, or to a mailing list.

2.1. Commands

2.1.1. Changing font style

Table 1 on the next page shows obsolete and ‘proper’ commands in $\text{\LaTeX 2}_{\epsilon}$ side by side for changing font style. Macros called “local” only apply to their own argument whereas those

⁶Of course you may change these values according to taste, but beware of fiddling with `\emergencystretch`.

Otherwise you’ll get quite sloppy justified text as you would get with a rather well-known text processor.

⁷The posting may be found as Message-ID: <a84us0\$plqcm\$7@ID-30533.news.dfncis.de>

2. Some obsolete commands and packages

Table 1: Commands for changing font style

obsolete	Replacement in L ^A T _E X 2 _ε	
	local	global/switch
<code>{\bf ...}</code>	<code>\textbf{...}</code>	<code>\bfseries</code>
<code>{\em ...}</code>	<code>\emph{...}</code>	<code>\em^a</code>
<code>{\it ...}</code>	<code>\textit{...}</code>	<code>\itshape</code>
—	<code>\textmd{...}</code>	<code>\mdseries</code>
<code>{\rm ...}</code>	<code>\textrm{...}</code>	<code>\rmfamily</code>
<code>{\sc ...}</code>	<code>\textsc{...}</code>	<code>\scshape</code>
<code>{\sf ...}</code>	<code>\textsf{...}</code>	<code>\sffamily</code>
<code>{\sl ...}</code>	<code>\textsl{...}</code>	<code>\slshape</code>
<code>{\tt ...}</code>	<code>\texttt{...}</code>	<code>\ttfamily</code>
—	<code>\textup{...}</code>	<code>\upshape</code>

^aMay be useful when defining macros.

called “global/switch” will apply to the whole following text to the end.

Why not use obsolete commands? Obsolete commands do not support L^AT_EX 2_ε’s new font selection scheme, or NFSS. `{\bf foo}`, for instance, resets all font attributes which had been set earlier before it prints *foo* in bold face. This is why you cannot simply define a bold-italics style by `{\it \bf Test}` only. (This definition will produce: **Test**.) On the other hand, the new commands `\textbf{\textit{Test}}` will behave as expected producing: ***Test***. Apart from that, with the former commands there is no “italics correction”, cf. for instance *halfhearted* (`{\it half}hearted`) to *halfhearted* (`\textit{half}hearted`).

2.1.2. Mathematical fractions (`\over` vs. `\frac`)

The `\over` command should be avoided. `\over` is a T_EX command which is even more complicated to parse or which cannot be parsed at all due to the syntax differing from L^AT_EX. The *amsmath.sty* package even redefines `\frac{}{}{}` which will result in error messages when using `\over`. Another point in favour of using `\frac{}{}{}` is that it is easier to fill in both the fraction’s numerator and denominator, especially with more complex fractions.

Replace: `$a \over b$` by `$(\frac{a}{b})$`

2.1.3. Centring text using `\centerline`

The `\centerline` command is another T_EX command which should be avoided in L^AT_EX. On the one hand `\centerline` is incompatible with some L^AT_EX packages, such as *color.sty*. On the other hand it may yield results you would rather like to avoid, or that you would not expect. For instance:

2. Some obsolete commands and packages

```
\begin{enumerate}  
\item \centerline{An item}  
\end{enumerate}
```

1.	An item
----	---------

Replace: `\centerline{...}` by `{\centering ...}`
or
`\begin{center}`
`...`
`\end{center}`

Note: On how to centre graphics and tables see section 3.1 on page 14.

2.2. Class files and packages

2.2.1. *scrlettr.cls* vs. *scrlettr2.cls*

scrlettr.cls class from the KOMA-Script bundle is obsolete. It was replaced by *scrlettr2.cls*. In order to produce a layout *similar* to the former KOMA-Script letter class use class option `KOMAold` which provides a compatibility mode.

Replace: `\documentclass{scrlettr}` by `\documentclass[KOMAold]{scrlettr2}`

Note: For new templates and letters use the new interface. It is definitely more flexible.

It is not possible to elaborate on the differences between the two user interfaces in this overview. See scrquien [4] for details.

2.2.2. *epsf.sty*, *psfig.sty*, *epsfig.sty* vs. *graphics.sty*, *graphicx.sty*

The *epsf.sty* and the *psfig.sty* packages have been replaced by *graphics.sty* and *graphicx.sty*. *epsfig.sty* is just a wrapper⁸ for processing old documents which had been done using *psfig.sty* with the *graphicx.sty* package.

As *epsfig.sty* uses *graphicx.sty* internally *epsfig.sty* still *may* be used. It should, however, not be used for new documents. For these *graphics.sty* or *graphicx.sty* should be preferred. *epsfig.sty* is mainly provided for reasons of compatibility, as mentioned above.

For the differences between *graphics.sty*, and *graphicx.sty* see grfguide [2]. For hints on centring graphics see section 3.1 on page 14.

Replace: `\usepackage{psfig}` by `\usepackage{graphicx}`
`\psfig{file=image,...}` `\includegraphics[...]{image}`

2.2.3. *doublestpace.sty* vs. *setstpace.sty*

For changing space between lines use the *setstpace.sty* package. *doublestpace.sty* is obsolete. It was replaced by *setstpace.sty*. Cf. section 1.4 on page 5.

Replace: `\usepackage{doublestpace}` by `\usepackage{setstpace}`

⁸A “wrapper” here means a style file which itself loads another one or more style files, hence modelling functions.

2. Some obsolete commands and packages

2.2.4. *fancyheadings.sty*, *scrpage.sty* vs. *fancyhdr.sty*, *scrpage2.sty*

The *fancyheadings.sty* package was replaced by *fancyhdr.sty*. Another way to modify headings is provided by the *scrpage2.sty* package from the KOMA-Script bundle. Note again not to use *scrpage.sty*. For documentation on *scrpage2.sty* see scrguien [4].

Replace: `\usepackage{fancyheadings}` by `\usepackage{fancyhdr}`

Replace: `\usepackage{scrpage}` by `\usepackage{scrpage2}`

2.2.5. *isolatin.sty*, *umlaut.sty* vs. *inputenc.sty*

Some general notes: Basically there are four ways to input German *umlauts* and other non-ASCII characters:

1. `H{"u}lle`: This will work on any given system anytime.

The main disadvantages, however, are that kerning⁹ between letters is disturbed badly; it is extremely complicated to input at least in a German-language text; and it is rather hard to read in source code.

So this variant should *always* be avoided due to the problems as far as kerning is concerned.

2. With `H"ulle` or `H{"u}lle` the abovementioned problems as far as kerning is concerned do not appear. It can be used on every system, too.

However it is just as tricky to input and to read the text as with the above variant.

This variant does make sense, however, when defining macros or style files for it does not require a particular text file encoding nor any additional packages.

3. With *(n)german.sty* or the `(n)german` option in *babel.sty* German *umlauts* can be input easier (`H"ulle`). Again this will work on all systems. As both *babel.sty* and *(n)german.sty* are available on all \TeX systems there should be no problems as far as compatibility is concerned.

However, this again is tricky to input, and the source is comparatively hard to read.

This variant is best for use in continuous text. But it should be avoided in macro definitions and in preambles.

4. Direct input (`Hü lle`). The advantage of this variant is obvious. You can input and read the continuous source text just as any other “normal” text.

On the other hand you have to tell \LaTeX which input encoding is used. There may also be problems when exchanging files between different systems. This is *not* a problem for \TeX , or \LaTeX itself, but it may cause *problems in displaying text in editors* on different systems. For instance, a Euro currency symbol encoded in iso-8859-15 (latin9) may be *displayed* in an editor on a windows box (CP1252) as $\text{\text{€}}$.

⁹“Kerning” means including positive or negative space between characters depending on which characters are to be typeset.

2. Some obsolete commands and packages

This variant is quite good for continuous text. It should, however, be avoided in macro definitions and in preambles.

To sum it up, in macros, in preambles, and in style files `H\"ulle`, or `H\"{u}lle` should be used, while in the rest of the text you should either use `H"ulle`, or `H\"ulle`.

Input Encoding For telling \LaTeX about the input encoding employed in a file do *not* use the packages *isolatin1.sty*, *isolatin.sty*, or *umlaut.sty*! Those packages are either obsolete, or they are not available on every system.

It is correct to use *inputenc.sty*. There are four options available:

latin1/latin9 for Unix-like systems (latin1 also works on MS Windows and Mac OS X)

ansinew for MS Windows

applemac for the Macintosh¹⁰

cp850 for OS/2

Replace: `\usepackage{isolatin1}` by `\usepackage[latin1]{inputenc}`

Replace: `\usepackage{umlaut}` by `\usepackage[latin1]{inputenc}`

2.2.6. *t1enc.sty* vs. *fontenc.sty*

Generally speaking the topic has been dealt with sufficiently in both the De-TeX-FAQ [5, questions 5.3.2, 5.3.3, 10.1.10], and the UK FAQ [3, “Why use *fontenc* rather than *t1enc*”, question 290]. So all that remains to be said in short is that *t1enc.sty* is obsolete and hence should be replaced by *fontenc.sty*!

Replace: `\usepackage{t1enc}` by `\usepackage[T1]{fontenc}`

2.2.7. *natdin.bst* vs. *dinat.bst*

Style file *natdin.bst* was replaced by *dinat.bst*.

Replace: `\bibliographystyle{natdin}` by `\bibliographystyle{dinat}`

2.3. Fonts

“Fonts and \LaTeX ” is a source of enduring “happiness” and “joy” in `de.comp.text.tex`. Most discussions start with the question why fonts display so “fuzzy” in Adobe Acrobat® Reader. Most answers point to the *times.sty* or *pslatex.sty* packages. However, when using these packages completely different fonts are used.

For making “Computer Modern” fonts display just fine in *acroread* see De-TeX-FAQ [5, question 9.2.3], or UK FAQ [3, “The wrong type of fonts in PDF”, question 79].

¹⁰latin1 encoding is recommended for OS X users, too, as it is better fit for exchanging files cross-platform than applemac. If you do so you should, however, check the encoding settings of your editor first. In the long run you might like to switch to unicode, but please note that unicode support in *inputenc.sty* still is a work in progress at this point of time.

2. Some obsolete commands and packages

2.3.1. *times.sty*

The *times.sty* package is obsolete (see psnfss2e [7]). It does set `\rmdefault` to the “Times” font, `\sfdefault` to “Helvetica”, and `\ttdefault` to “Courier”. However it does *not* use the appropriate mathematical fonts. What’s more, Helvetica is not scaled correctly which makes it appear too big. So if you want to use the combination Times/Helvetica/Courier you should do this:

Replace: `\usepackage{times}` by `\usepackage{mathptmx}`
`\usepackage[scaled=.90]{helvet}`
`\usepackage{courier}`

Note. The scaling factor for *helvet.sty* combined with Times should be somewhere between 0.90 and 0.92.

2.3.2. *mathptm.sty*

mathptm.sty is the predecessor to *mathptmx.sty*.

Replace: `\usepackage{mathptm}` by `\usepackage{mathptmx}`

2.3.3. *pslatex.sty*

The *pslatex.sty* package internally works like *mathptm.sty* + *helvet.sty* (scaled). However, it uses a Courier scaled too narrowly. The main disadvantage with using *pslatex.sty* is that it does *not* work with T1 and TS1 encodings.

Replace: `\usepackage{pslatex}` by `\usepackage{mathptmx}`
`\usepackage[scaled=.90]{helvet}`
`\usepackage{courier}`

Note on all combinations of Times/Helvetica You may use the usual `cmtt` font for type-writer faces, not loading the *courier.sty* package at all.

2.3.4. *palatino.sty*

The *palatino.sty* package behaves like *times.sty* – apart from setting `\rmdefault` to “Palatino”, of course. This is why it should not be used any more.

Replace: `\usepackage{palatino}` by `\usepackage{mathpazo}`
`\usepackage[scaled=.95]{helvet}`
`\usepackage{courier}`

Note: The scaling factor for *helvet.sty* in combination with the Palatino font should be set to 0.95.

“Helvetica” is *not* the best sans-serif font there is for use with “Palatino”. It rather is the best *freely-available* one. He that possesses a CorelDraw®-CD (this may well be an older version)

2. Some obsolete commands and packages

can use “Palatino” quite well along with “Frutiger”¹¹, or “Optima”¹². Walter Schmidt provides the adaptations for using some PostScript fonts with T_EX on his homepage.¹³

2.3.5. *mathpple.sty*

This package used to be the predecessor to *mathpazo.sty*. Some symbols are lacking, so those fonts are taken from the Euler fonts instead. Some other symbols are not fit for use with Palatino as the font metrics are not correct. For details cf. psnfss2e [7].

2.3.6. Typesetting upright greek letters

The passages I have marked as red in the following are not obsolete in the sense of “you should not use this any more”, but now editing text is made much easier by the *upgreek.sty* package. For some more hints on usage please see the documentation upgreek [6].

The *pifont.sty* tricks

Replace:	by
<code>\usepackage{pifont}</code>	<code>\usepackage{upgreek}</code>
<code>\newcommand{\uppi}{\Pisymbol{psy}{112}}</code>	<code>\$\uppi\$</code>
<code>\uppi</code>	

or

```
\newcommand[1]{\upgreek}{%
  \usefont{U}{psy}{m}{n}#1}
\upgreek{p}
```

The *babel.sty* trick

Replace:	by
<code>\usepackage[greek,...]{babel}</code>	<code>\usepackage{upgreek}</code>
<code>\newcommand[1]{\upgreek}{%</code>	<code>\$\uppi\$</code>
<code>\foreignlanguage{greek}{#1}}</code>	
<code>\upgreek{p}</code>	

2.3.7. *euler.sty*

You should use *eulervm.sty* instead of *euler.sty* for mathematical typesetting. *eulervm.sty* is a L^AT_EX package for using the eulervm fonts. These are virtual math fonts based on both the Euler and the CM fonts, consuming less of T_EX’s resources and supplying some improved math symbols. A usable `\hslash` is also supplied.

Replace: `\usepackage{euler}` by `\usepackage{eulervm}`

¹¹Bitstream “Humanist 777”, bfr

¹²Bitstream “Zapf Humanist”, bop

¹³Fonts for T_EX: <http://home.vr-web.de/was/fonts>

3. Miscellaneous

This section – apart from 3.2 on the next page – gives some more general advice than the “deadly sins” section.

3.1. Floats – “figure”, “table”

For centring a float environment you should use `\centering` instead of `\begin{center}`-`\end{center}` because the latter will include an additional vertical skip you will rather want to avoid in most cases.

Replace:

<code>\begin{figure}</code>	by <code>\begin{figure}</code>
<code>\begin{center}</code>	<code>\centering</code>
<code>\includegraphics{bild}</code>	<code>\includegraphics{bild}</code>
<code>\end{center}</code>	<code>\end{figure}</code>
<code>\end{figure}</code>	

Note: However, when centring a region within continuous text or within a `titlepage`-environment this additional space may be quite welcome.

3.2. The appendix

The appendix is introduced by the *switch* `\appendix`. This is *not* an environment.

Replace:

<code>\begin{appendix}</code>	by <code>\appendix</code>
<code>\section{Blub}</code>	<code>\section{Blub}</code>
<code>\end{appendix}</code>	

3.3. Mathematical typesetting

Generally speaking, you should use *amsmath.sty* for advanced mathematical typesetting, providing a number of new environments replacing `eqnarray` in the first place. The main advantages of the package are these:

- Spacing within and around environments is more consistent.
- Equation numbering will be placed in a way so that they will not be printed over any more.
- Some new environments, e.g. `split`, provide a solution to split up long equations easily.
- It is easy to define new operators (similar to `\sin` etc.) with proper spacing.

Warning: When using the *amsmath.sty* package you should *never* use the `displaymath`, `eqnarray`, or `eqnarray*` environments for these are not supported by *amsmath.sty*. Otherwise this would lead to inconsistent spacing.

`\[...\]` is adapted correctly by *amsmath.sty*. So it may be used instead of `displaymath`, `eqnarray`, and `eqnarray*` may be replaced by `align`, or `align*`. For a complete overview of *amsmath.sty* see `amslatex` [1].

3. Miscellaneous

Replace: <code>\begin{eqnarray}</code>	by <code>\begin{align}</code>
<code>a &=& b \\\</code>	<code>a &= b \\\</code>
<code>b &=& c \\\</code>	<code>b &= c \\\</code>
<code>a &=& c</code>	<code>a &= c</code>
<code>\end{eqnarray}</code>	<code>\end{align}</code>

3.4. Language-specific macros – `*name`

From time to time the question comes up in `de.comp.text.tex` how to modify, for instance, “References” to “Literaturliste” or to some other heading. So I have compiled those macros in table 2. They have been taken from the *german.sty* package. Users who want to adapt macro output to other languages may as well refer to this table as an example.

For instance, if you want to change the “List of Figures” which usually reads “Abbildungsverzeichnis” in German to, say, “Abbildungen” you may use the following command:

```
\renewcommand*{\listfigurename}{Abbildungen}
```

The other macros are changed in the same way respectively. When using *babel.sty* use the `\addto` macro. For more details see the De-TeX-FAQ [5].

```
\addto{\captionsngerman}{%  
  \renewcommand*{\listfigurename}{Bilderliste}}
```

3. Miscellaneous

Table 2: *Macros defined by (n)german.sty or by babel.sty with the (n)german option*

Name of macro	Original definition	Usual output in German
\prefacename	Preface	Vorwort
\refname ^a	References	Literatur
\abstractname	Abstract	Zusammenfassung
\bibname ^b	Bibliography	Literaturverzeichnis
\chaptername	Chapter	Kapitel
\appendixname	Appendix	Anhang
\contentsname	Contents	Inhaltsverzeichnis
\listfigurename	List of Figures	Abbildungsverzeichnis
\listtablename	List of Tables	Tabellenverzeichnis
\indexname	Index	Index
\figurename	Figure	Abbildung
\tablename	Table	Tabelle
\partname	Part	Teil
\enclname	encl	Anlage(n)
\ccname	cc	Verteiler
\headtoname	To	An
\pagename	Page	Seite
\seename	see	siehe
\alsoname	see also	siehe auch

^aIn article class only.

^bIn report and book classes only.

A. An example illustrating the effect of the \sloppy command

This is the example Markus Kohm published:

```
\documentclass{article}

\setlength{\textwidth}{20em}
\setlength{\parindent}{0pt}
\begin{document}
\typeout{First without \string\sloppy\space and underfull \string\hbox}

tatata tatata tatata tatata tatata tatata ta\~ta\~tata
tatata tatata tatata tatata tatata tatata tata\~tata
tatata tatata tatata tatata ta\~tatatatt\~ta
tatata tatata tatata tatata tatata tatata ta\~ta\~ta\~ta

\typeout{done.}

\sloppy
\typeout{Second with \string\sloppy\space and underfull \string\hbox}

tatata tatata tatata tatata tatata tatata ta\~ta\~tata
tatata tatata tatata tatata tatata tatata tata\~tata
tatata tatata tatata tatata ta\~tatatatt\~ta
tatata tatata tatata tatata tatata tatata ta\~ta\~ta\~ta

\typeout{done.}
\end{document}
```

Quelle: Message-ID: <8557097.gEimXdBtjU@ID-107054.user.dfncis.de>

References

- [1] AMERICAN MATHEMATICAL SOCIETY: *User's Guide for the amsmath Package*. December 1999, Version 2.0.
URL: CTAN:macros/latex/required/amslatex/.
- [2] DAVID P. CARLISLE: *Packages in the 'graphics' bundle*. January 1999.
URL: CTAN:macros/latex/required/graphics/.
- [3] ROBIN FAIRBAIRNS: *The UK T_EX FAQ. Your 322 Questions Answered*. WWW, Version 3.10a, 07 November 2003,
URL: <http://www.tex.ac.uk/faq>.
- [4] MARKUS KOHM, FRANK NEUKAM und AXEL KIELHORN: *The KOMA-Script Bundle*. scruien.
URL: CTAN:macros/latex/supported/koma-script/.
- [5] BERND RAICHLE, ROLF NIEPRASCHK und THOMAS HAFNER: *Fragen und Antworten (FAQ) über das Textsatzsystem T_EX und DANTE, Deutschsprachige Anwendervereinigung T_EX e.V.* WWW, Version 72. September 2003,
URL: <http://www.dante.de/faq/de-tex-faq/>.
- [6] WALTER SCHMIDT: *The upgreek package for L^AT_EX 2_ε*. May 2001, Version 1.0.
URL: CTAN:macros/latex/contrib/supported/was/.
- [7] WALTER SCHMIDT: *Using common PostScript fonts with L^AT_EX*. April 2002, PSNFSS version 9.0.
URL: CTAN:macros/latex/required/psnfss/psnfss2e.pdf
- [8] WALTER SCHMIDT, JÖRG KNAPPEN, HUBERT PARTL und IRENE HYNÄ: *L^AT_EX 2_ε-Kurzbeschreibung*. April 1999, Version 2.1.
URL: CTAN:info/lshort/german/. English Translation available at
URL: CTAN:info/lshort/english/