

Grzegorz Murzynowski

# The gmutils Package\*

Written by Grzegorz Murzynowski,  
natror at o2 dot pl

© 2005, 2006, 2007, 2008 by Grzegorz Murzynowski.

This program is subject to the L<sup>A</sup>T<sub>E</sub>X Project Public License.

See <http://www.ctan.org/tex-archive/help/Catalogue/licenses.lpl.html>  
for the details of that license.

LPPL status: "author-maintained".

Many thanks to my T<sub>E</sub>X Guru Marcin Woliński for his T<sub>E</sub>Xnical support.

```
76 \NeedsTeXFormat{LaTeX2e}
77 \ProvidesPackage{gmutils}
78 [2008/08/07_v0.92_some_rather_TeXnical_macros,_some_of_them_
    tricky_(GM)]
```

## Contents

<b>Intro</b> . . . . .	2	Negative \addvspace . . . . .	22
Installation . . . . .	2	My heading setup for mwcls . . . . .	24
Contents of the gmutils.zip Archive . . . . .	2	<b>Compatibilising Standard and mwcls</b>	
Compiling of the Documentation . . . . .	2	<b>Sectionings</b> . . . . .	25
<b>A couple of abbreviations</b> . . . . .	3	<b>enumerate* and itemize*</b> . . . . .	26
\@ifnextcat, \@ifnextac . . . . .	5	<b>The Logos</b> . . . . .	27
\afterfi and Pals . . . . .	7	<b>Expanding turning stuff all into ‘other’</b> . . . . .	29
<b>Environments redefined</b> . . . . .	7	<b>Brave New World of X<sub>3</sub>TeX</b> . . . . .	30
Almost an Environment or		Fractions . . . . .	30
Redefinition of \begin . . . . .	7	\resizegraphics . . . . .	31
\@ifenvir and Improvement of \end . . . . .	8	<b>Varia</b> . . . . .	32
<b>From relsize</b> . . . . .	9	\@ifempty . . . . .	36
\firstofone and the Queer \catcodes . . . . .	10	\include not only .tex’s . . . . .	37
<b>Some ‘other’ stuff</b> . . . . .	10	Faked small caps . . . . .	38
<b>Metasymbols</b> . . . . .	11	See above/see below . . . . .	39
<b>Macros for Printing Macros and</b>		luzniej and napa-	
<b>Filenames</b> . . . . .	12	pierki—environments	
<b>Storing and Restoring the Meanings</b>		used in page breaking for money . . . . .	39
<b>of CSs</b> . . . . .	14	<b>Settings for mathematics in main font</b> . . . . .	42
<b>Not only preamble!</b> . . . . .	17	<b>Typesetting dates in my memoirs</b> . . . . .	45
<b>Third Person Pronouns</b> . . . . .	17	Minion and Garamond Premier	
<b>To Save Precious Count Registers</b> . . . . .	18	kerning and ligature fixes . . . . .	48
<b>Improvements to mwcls Sectioning</b>		<b>Change History</b> . . . . .	49
<b>Commands</b> . . . . .	19	<b>Index</b> . . . . .	50
An improvement of MW’s			
\SetSectionFormatting . . . . .	21		

\* This file has version number v0.92 dated 2008/08/07.

## Intro

The `gmutils.sty` package provides some macros that are analogous to the standard L<sup>A</sup>T<sub>E</sub>X ones but extend their functionality, such as `\@ifnextcat`, `\addtomacro` or `\begin(*)`. The others are just conveniences I like to use in all my TeX works, such as `\afterfi`, `\pk` or `\cs`.

I wouldn't say they are only for the package writers but I assume some nonzero (L<sup>A</sup>)T<sub>E</sub>X-awareness of the user.

For details just read the code part.

## Installation

Unpack the `gmutils-tds.zip` archive (this is an archive that conforms the TDS standard, see CTAN/tds/tds.pdf) in some `texmf` directory or just put the `gmutils.sty` somewhere in the `texmf/tex/latex` branch. Creating a `texmf/tex/latex/gm` directory may be advisable if you consider using other packages written by me.

Then you should refresh your T<sub>E</sub>X distribution's files' database most probably.

## Contents of the gmutils.zip Archive

The distribution of the `gmutils` package consists of the following four files and a TDS-compliant archive.

```
gmutils.sty
README
gmutilsDoc.tex
gmutilsDoc.pdf
gmutils.tds.zip
```

## Compiling of the Documentation

The last of the above files (the `.pdf`, i.e., *this file*) is a documentation compiled from the `.sty` file by running L<sup>A</sup>T<sub>E</sub>X on the `gmutilsDoc.tex` file twice (`xelatex gmutils.sty` in the directory you wish the documentation to be in, you don't have copy the `.sty` file there, T<sub>E</sub>X will find it), then `MakeIndex` on the `gmutils.idx` file, and then L<sup>A</sup>T<sub>E</sub>X on `gmutilsDoc.tex` once more.

`MakeIndex` shell command:

```
makeindex -r gmutilsDoc
```

The `-r` switch is to forbid `MakeIndex` to make implicit ranges since the (code line) numbers will be hyperlinks.

Compiling the documentation requires the packages: `gmdoc` (`gmdoc.sty` and `gmdocc.cls`), `gmverb.sty`, `gmutils.sty`, `gmiflink.sty` and also some standard packages: `hyperref.sty`, `color.sty`, `geometry.sty`, `multicol.sty`, `lmodern.sty`, `fontenc.sty` that should be installed on your computer by default.

If you had not installed the `mwcls` classes (available on CTAN and present in T<sub>E</sub>X Live e.g.), the result of your compilation might differ a bit from the `.pdf` provided in this `.zip` archive in formatting: If you had not installed `mwcls`, the standard `article.cls` class would be used.

```
151 \ifx\XeTeXversion\relax
152 \let\XeTeXversion\@undefined% If someone earlier used the \@ifundefined{%
    XeTeXversion} to test whether the engine is XETEX, then \XeTeXversion is
    defined in the sense of  $\varepsilon$ -TEX tests. In that case we \let it to something really
    undefined. Well, we might keep sticking to \@ifundefined, but it's a macro
```

and it eats its arguments, freezing their catcodes, which is not what we want in line 2788

```

159 \fi
161 \ifdefined\XeTeXversion
162 \XeTeXinputencoding_\utf-8% we use Unicode dashes later in this file.
163 \fi% and if we are not in XeTeX, we skip them thanks to XeTeX-test.

```

## A couple of abbreviations

```

\@xa 169 \let\@xa\expandafter
\@nx 170 \let\@nx\noexpand

```

The `\newgif` declaration's effect is used even in the  $\text{\LaTeX 2}_\epsilon$  source by redefining some particular user defined ifs (UD-ifs henceforth) step by step. The goal is to make the UD-if's assignment global. I needed it at least twice during `gmdoc` writing so I make it a macro. It's an almost verbatim copy of  $\text{\LaTeX}$ 's `\newif` modulo the letter *g* and the `\global` prefix. (File `d:ltdefs.dtx` Date: 2004/02/20 Version v1.3g, lines 139–150)

```

\newgif 181 \protected\def\newgif#1{%
182   {\escapechar\m@ne
183     \global\let#1\iffalse
184     \@gif#1\iftrue
185     \@gif#1\iffalse
186   }}

```

'Almost' is also in the detail that in this case, which deals with `\global` assignments, we don't have to bother with storing and restoring the value of `\escapechar`: we can do all the work inside a group.

```

\@gif 192 \def\@gif#1#2{%
193   \protected\@xa\gdef\csname\@xa\@gobbletwo\string#1%
194     g% the letter g for '\global'.
195     \@xa\@gobbletwo\string#2\endcsname
196   {\global\let#1#2}}

```

```

198 \protected\def\newif#1{% We not only make \newif \protected but also make
    it to define \protected assignments so that premature expansion doesn't
    affect \if...\fi nesting.

```

```

205   \count@\escapechar_\escapechar\m@ne
206   \let#1\iffalse
207   \@if#1\iftrue
208   \@if#1\iffalse
209   \escapechar\count@}

```

```

\@if 211 \def\@if#1#2{%
212   \protected_\@xa\def\csname\@xa\@gobbletwo\string#1%
213     \@xa\@gobbletwo\string#2\endcsname
214   {\let#1#2}}

```

After `\newgif\ifffoo` you may type `{\foogtrue}` and the `\ifffoo` switch becomes globally equal `\iftrue`. Simili modo `\foogfalse`. Note the letter *g* added to underline globalness of the assignment.

If for any reason, no matter how queer ;-) may it be, you need *both* global and local switchers of your `\if...`, declare it both with `\newif` and `\newgif`.

Note that it's just a shorthand. `\global\if<switch>true/false` *does* work as expected.

There's a trouble with `\refstepcounter`: defining `\@currentlabel` is local. So let's `\def` a `\global` version of `\refstepcounter`.

Warning. I use it because of very special reasons in `gmdoc` and in general it is probably not a good idea to make `\refstepcounter` global since it is contrary to the original L<sup>A</sup>T<sub>E</sub>X approach.

```
\grefstepcounter 236 \protected\def\grefstepcounter#1{%
237   {\let\protected@edef=\protected@xdef\refstepcounter{#1}}}
```

Naïve first try `\globaldefs=\tw@` raised an error unknown command `\reserved@e`. The matter was to globalize `\protected@edef` of `\@currentlabel`.

Thanks to using the true `\refstepcounter` inside, it observes the change made to `\refstepcounter` by `hyperref`.

2008/08/10 I spent all the night debugging `\penalty 10000` that was added after a `hypertarget` in vertical mode. I didn't dare to touch `hyperref`'s guts, so I worked it around with ensuring every `\grefstepcounter` to be in `hmode`:

```
\hgrefstepcounter 251 \protected\def\hgrefstepcounter#1{%
252   \ifhmode\leavevmode\fi\grefstepcounter{#1}}
```

By the way I read some lines from *The T<sub>E</sub>Xbook* and was reminded that `\unskip` strips any last skip, whether horizontal or vertical. And I use `\unskip` mostly to replace a blank space with some fixed skip. Therefore define

```
\hunskip 259 \protected\def\hunskip{\ifhmode\unskip\fi}
```

Note the two macros defined above are `\protected`. I think it's a good idea to make `\protected` all the macros that contain assignments. There is one more thing with `\ifhmode`: it can be different at the point of `\edef` and at the point of execution.

Another shorthand. It may decrease a number of `\expandafters` e.g.

```
\glet 269 \def\glet{\global\let}
```

L<sup>A</sup>T<sub>E</sub>X provides a very useful `\g@addto@macro` macro that adds its second argument to the current definition of its first argument (works iff the first argument is a no argument macro). But I needed it some times in a document, where `@` is not a letter. So:

```
\gaddtomacro 277 \let\gaddtomacro=\g@addto@macro
```

The redefining of the first argument of the above macro(s) is `\global`. What if we want it local? Here we are:

```
\addto@macro 282 \long\def\addto@macro#1#2{%
283   \toks@{\@xa{#1#2}}%
284   \edef#1{\the\toks@}%
285 }% (\toks@ is a scratch register, namely \tokso.)
```

And for use in the very document,

```
\addtomacro 289 \let\addtomacro=\addto@macro
```

2008/08/09 I need to prepend something not add at the end—so

```
\prependtomacro 292 \long\def\prependtomacro#1#2{%
293   \edef#2{\unexpanded{#1}\@xa\unexpanded\@xa{#2}}}
```

Note that `\prependtomacro` can be prefixed.

```
\addtotoks 297 \long\def\addtotoks#1#2{%
298   #1=\@xa{\the#1#2}}
```

```
\@emptyify 301 \newcommand*\@emptyify[1]{\let#1=\@empty}
\emptyify 302 \@ifdefinable\emptyify{\let\emptyify\@emptyify}
```

Note the two following commands are in fact one-argument.

```
\g@emptyify 306 \newcommand*\g@emptyify{\global\@emptyify}
\gemptyify 307 \@ifdefinable\gemptyify{\let\gemptyify\g@emptyify}

\@relaxen 310 \newcommand\@relaxen[1]{\let#1=\relax}
\relaxen 311 \@ifdefinable\relaxen{\let\relaxen\@relaxen}
```

Note the two following commands are in fact one-argument.

```
\g@relaxen 315 \newcommand*\g@relaxen{\global\@relaxen}
\grelaxen 316 \@ifdefinable\grelaxen{\let\grelaxen\g@relaxen}
```

For the heavy debugs I was doing while preparing gmdoc, as a last resort I used `\showlists`. But this command alone was usually too little: usually it needed setting `\showboxdepth` and `\showboxbreadth` to some positive values. So,

```
\gmshowlists 326 \def\gmshowlists{\showboxdepth=1000\showboxbreadth=1000\%
\showlists}
```

```
\nameshow 329 \newcommand\nameshow[1]{\@xa\show\csname#1\endcsname}
\nameshowthe 330 \newcommand\nameshowthe[1]{\@xa\showthe\csname#1\endcsname}
```

Note that to get proper `\showthe\my@dimen14` in the ‘other’ `@`’s scope you write `\nameshowthe{my@dimen}14`.

Standard `\string` command returns a string of ‘other’ chars except for the space, for which it returns `_10`. In gmdoc I needed the spaces in macros’ and environments’ names to be always `_12`, so I define

```
\xiistring 341 \def\xiistring#1{%
342 \if\@nx#1\xiispace
343 \xiispace
344 \else
345 \string#1%
346 \fi}
```

`\@ifnextcat`, `\@ifnextac`

As you guess, we `\def \@ifnextcat` à la `\@ifnextchar`, see L<sup>A</sup>T<sub>E</sub>X<sub>2 $\epsilon$</sub>  source dated 2003/12/01, file `d`, lines 253–271. The difference is in the kind of test used: while `\@ifnextchar` does `\ifx`, `\@ifnextcat` does `\ifcat` which means it looks not at the meaning of a token(s) but at their `\catcode`(s). As you (should) remember from *The T<sub>E</sub>Xbook*, the former test doesn’t expand macros while the latter does. But in `\@ifnextcat` the peeked token is protected against expanding by `\noexpand`. Note that the first parameter is not protected and therefore it shall be expanded if it’s a macro. Because an assignment is involved, you can’t test whether the next token is an active char.

```
\@ifnextcat 363 \long\def\@ifnextcat#1#2#3{%
367 \def\reserved@d{#1}%
368 \def\reserved@a{#2}%
369 \def\reserved@b{#3}%
370 \futurelet\@let@token\@ifncat}

\@ifncat 373 \def\@ifncat{%
374 \ifx\@let@token\@sptoken
375 \let\reserved@c\@xifncat
376 \else
377 \ifcat\reserved@d\@nx\@let@token
```

```

378     \let\reserved@c\reserved@a
379     \else
380     \let\reserved@c\reserved@b
381     \fi
382     \fi
383     \reserved@c}
385 {\def\:{\let\@sptoken= }\:}% this makes \@sptoken a space token.
388 \def\:{\@xifncat}\@xa\gdef\:{\futurelet\@let@token\@ifncat}}

```

Note the trick to get a macro with no parameter and requiring a space after it. We do it inside a group not to spoil the general meaning of \: (which we extend later).

The next command provides the real \if test for the next token. *It* should be called \@ifnextchar but that name is assigned for the future \ifx text, as we know. Therefore we call it \@ifnextif.

```

\@ifnextif 399 \long\def\@ifnextif#1#2#3{%
403     \def\reserved@d{#1}%
404     \def\reserved@a{#2}%
405     \def\reserved@b{#3}%
406     \futurelet\@let@token\@ifnif}

\@ifnif 409 \def\@ifnif{%
410     \ifx\@let@token\@sptoken
411     \let\reserved@c\@xifnif
412     \else
413     \if\reserved@d\@nx\@let@token
414     \let\reserved@c\reserved@a
415     \else
416     \let\reserved@c\reserved@b
417     \fi
418     \fi
419     \reserved@c}

422 {\def\:{\let\@sptoken= }\:}% this makes \@sptoken a space
      token.
424 \def\:{\@xifnif}\@xa\gdef\:{\futurelet\@let@token\@ifnif}}

```

But how to peek at the next token to check whether it's an active char? First, we look with \@ifnextcat whether there stands a group opener. We do that to avoid taking a whole {...} as the argument of the next macro, that doesn't use \futurelet but takes the next token as an argument, tests it and puts back intact.

```

\@ifnextcat 436 \long\def\@ifnextcat#1#2{%
437     \@ifnextcat\bgroup{#2}{\gm@ifnac{#1}{#2}}}}

\gm@ifnac 439 \long\def\gm@ifnac#1#2#3{%
440     \ifcat\@nx~\@nx#3\afterfi{#1#3}\else\afterfi{#2#3}\fi}

```

Yes, it won't work for an active char \let to {<sub>1</sub>, but it *will* work for an active char \let to a char of catcode ≠ 1. (Is there anybody on Earth who'd make an active char working as \bgroup?)

Now, define a test that checks whether the next token is a genuine space, <sub>10</sub> that is. First define a CS let such a space. The assignment needs a little trick (*The T<sub>E</sub>Xbook* appendix D) since \let's syntax includes one optional space after =.

```

452 \let\gmu@reserveda\*%

```

```

\* 453 \def\*{%
454   \let\*\gmu@reserveda
455   \let\gm@letspace=\}%
456 \*_{%
@ifnextspace 459 \def\@ifnextspace#1#2{%
460   \let\gmu@reserveda\*%
\* 461   \def\*{%
462     \let\*\gmu@reserveda
463     \ifx\@let@token\gm@letspace\afterfi{#1}%
464     \else\afterfi{#2}%
465     \fi}%
466   \futurelet\@let@token\*}

```

First use of this macro is for an active – that expands to --- if followed by a space. Another to make dot checking whether is followed by ~ without gobbling the space if it occurs instead.

## \afterfi and Pals

It happens from time to time that you have some sequence of macros in an \if... and you would like to expand \fi before expanding them (e.g., when the macros should take some tokens next to \fi... as their arguments. If you know how many macros are there, you may type a couple of \expandafters and not to care how terrible it looks. But if you don't know how many tokens will there be, you seem to be in a real trouble. There's the Knuthian trick with \next. And here another, revealed to me by my T<sub>E</sub>X Guru.

I think the situations when the Knuthian (the former) trick is not available are rather seldom, but they are imaginable at least: the \next trick involves an assignment so it won't work e.g. in \edef. But in general it's only a matter of taste which one to use.

One warning: those macros peel the braces off, i.e.,

```
\if...\afterfi{\@makeother\^M}\fi
```

causes a leakage of  $\text{^M}_{12}$ . To avoid pollution write

```
\if...\afterfi{\bgroup\@makeother\^M\egroup}\fi.
```

```
\afterfi 497 \long\def\afterfi#1#2\fi{\fi#1}
```

And two more of that family:

```
\afterfifi 499 \long\def\afterfifi#1#2\fi#3\fi{\fi\fi#1}
```

```
\afteriffifi 500 \long\def\afteriffifi#1#2\if#3\fi#4\fi{\fi#1}
```

Notice the refined elegance of those macros, that cover both 'then' and 'else' cases thanks to #2 that is discarded.

```
\afteriffiffifi 504 \long\def\afteriffiffifi#1#2\fi#3\fi#4\fi{\fi#1}
```

```
\afteriffiffifi 505 \long\def\afteriffiffifi#1#2\fi#3\fi#4\fi{\fi\fi#1}
```

```
\afteriffiffifi 506 \long\def\afteriffiffifi#1#2\fi#3\fi#4\fi{\fi\fi\fi#1}
```

## Environments redefined

### Almost an Environment or Redefinition of \begin

We'll extend the functionality of \begin: the non-starred instances shall act as usual and we'll add the starred version. The difference of the latter will be that it won't check whether the 'environment' has been defined so any name will be allowed.

This is intended to structure the source with named groups that don't have to be especially defined and probably don't take any particular action except the scoping.

(If the `\begin*`'s argument is a (defined) environment's name, `\begin*` will act just like `\begin`.)

Original L<sup>A</sup>T<sub>E</sub>X's `\begin`:

```
\def\begin#1{%
  \ifundefined{#1}%
    {\def\reserved@a{\@latex@error{Environment #1
      undefined}\@eha}}%
    {\def\reserved@a{\def\@currenvir{#1}%
      \edef\@currenvline{\on@line}%
      \csname #1\endcsname}}%
  \@ignorefalse
  \begingroup\@endpefalse\reserved@a}
```

```
\@begnamedgroup 537 \long\def\@begnamedgroup#1{%
538   \@ignorefalse% not to ignore blanks after group
539   \begingroup\@endpefalse
540   \edef\@currenvir{#1}% We could do recatcoding through \string but all the
      name 'other' could affect a thousand packages so we don't do that and we'll
      recatcode in a testing macro, see line 590.
544   \edef\@currenvline{\on@line}%
545   \csname_#1\endcsname}% if the argument is a command's name (an environ-
      ment's e.g.), this command will now be executed. (If the corresponding
      control sequence hasn't been known to TEX, this line will act as \relax.)
```

For back compatibility with my earlier works

```
\bnamegroup 553 \let\bnamegroup\@begnamedgroup
```

And for the ending

```
\enamegroup 555 \def\enamegroup#1{\end{#1}}
```

And we make it the starred version of `\begin`.

```
\begin* 561 \def\begin{\@ifstar{\@begnamedgroup}{%
```

```
\begin 562   \@begnamedgroup@ifcs}}
```

```
\@begnamedgroup@ifcs 565 \def\@begnamedgroup@ifcs#1{%
566   \ifcsname#1\endcsname\afterfi{\@begnamedgroup{#1}}%
567   \else\afterfi{\@latex@error{Environment_#1_undefined}\@eha}%
568   \fi}%
```

## `\@ifenvir` and Improvement of `\end`

It's very clever and useful that `\end` checks whether its argument is ifx-equivalent `@currenvir`. However, in standard L<sup>A</sup>T<sub>E</sub>X it works not quite as I would expect: Since the idea of environment is to open a group and launch the cs named in the `\begin`'s argument. That last thing is done with `\csname... \endcsname` so the char catcodes are equivalent. Thus should be also in the `\end`'s test and therefore we ensure the compared texts are both expanded and made all 'other'.

First a (not expandable) macro that checks whether current environment is as given in #1.

```
\@ifenvir 590 \long\def\@ifenvir#1#2#3{%
592   \edef\gmu@reserveda{\@xa\string\csname\@currenvir\endcsname}%
593   \edef\gmu@reservedb{\@xa\string\csname#1\endcsname}%
```



```

594 \ifx\gmu@reserveda\gmu@reservedb\afterfi{#2}%
595 \else\afterfi{#3}%
596 \fi}
\@checkend 598 \def\@checkend#1{\@ifenvir{#1}{\@badend{#1}}{}}

```

Thanks to it you may write `\begin{macrocode*}` with  $*_{12}$  and end it with `\end{macrocode*}` with  $*_{11}$  (that was the problem that led me to this solution). The error messages looked really funny:

! LaTeX Error: `\begin{macrocode*}` on input line 1844 ended by `\end{macrocode*}`.

Of course, you might write also `\end{macrocode\star}` where `\star` is defined as 'other' star or letter star.

## From relsize

As file `relsize.sty`, v3.1 dated July 4, 2003 states, L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> version of these macros was written by Donald Arseneau [asnd@triumf.ca](mailto:asnd@triumf.ca) and Matt Swift [swift@bu.edu](mailto:swift@bu.edu) after the L<sup>A</sup>T<sub>E</sub>X 2.09 `smaller.sty` style file written by Bernie Cosell [cosell@WILMA.BBN.COM](mailto:cosell@WILMA.BBN.COM).

I take only the basic, non-math mode commands with the assumption that there are the predefined font sizes.

```

\relsize You declare the font size with \relsize{<n>} where <n> gives the number of steps
("mag-step" = factor of 1.2) to change the size by. E.g., n = 3 changes from \normalsize
\smaller to \LARGE size. Negative n selects smaller fonts. \smaller == \relsize{-1};
\larger \larger == \relsize{1}. \smallerr(my addition) == \relsize{-2}; \largerr
\smallerr guess yourself.
\largerr

```

(Since `\DeclareRobustCommand` doesn't issue an error if its argument has been defined and it only informs about redefining, loading `relsize` remains allowed.)

```

\relsize 636 \DeclareRobustCommand*\relsize[1]{%
637 \ifmmode\@nomath\relsize\else
638 \begingroup
639 \tempcnta% assign number representing current font size
640 \ifx\@currsize\normalsize_4\else_{}_{}% funny order is to have most
...
641 \ifx\@currsize\small_3\else_{}_{}_{}_{}_{}% ...likely sizes checked first
642 \ifx\@currsize\footnotesize_2\else
643 \ifx\@currsize\large_5\else
644 \ifx\@currsize\Large_6\else
645 \ifx\@currsize\LARGE_7\else
646 \ifx\@currsize\scriptsize_1\else
647 \ifx\@currsize\tiny_0\else
648 \ifx\@currsize\huge_8\else
649 \ifx\@currsize\Huge_9\else
650 4\rs@unknown@warning_{}% unknown state: \normalsize as
starting point
651 \fi\fi\fi\fi\fi\fi\fi\fi\fi

```

Change the number by the given increment:

```

653 \advance\tempcnta#1\relax

```

watch out for size underflow:

```

655 \ifnum\tempcnta<\z@_{} \rs@size@warning{small}{\string\tiny}%
\tempcnta\z@_{} \fi
656 \xa\endgroup

```

```

657         \ifcase\@tempcnta_\% set new size based on altered number
658         \tiny_\or_\scriptsize_\or_\footnotesize_\or_\small_\or_\%
            \normalsize_\or
659         \large_\or_\Large_\or_\LARGE_\or_\huge_\or_\Huge_\else
660         \rs@size@warning{large}{\string\Huge}\Huge
661 \fi\fi}% end of \relsize.

\rs@size@warning 664 \providecommand*\rs@size@warning[2]{\PackageWarning{gmutils_
            (relsize)}{%
665   Size requested is too #1. \MessageBreak Using #2 instead}}

\rs@unknown@warning 668 \providecommand*\rs@unknown@warning{\PackageWarning{gmutils_
            (relsize)}{Current font size
669   is unknown! (Why?!?) \MessageBreak Assuming \string\normalsize}}

And a handful of shorthands:

\larger 673 \DeclareRobustCommand*\larger[1][\@ne]{\relsize{+#1}}
\smaller 674 \DeclareRobustCommand*\smaller[1][\@ne]{\relsize{-#1}}
\textlarger 675 \DeclareRobustCommand*\textlarger[2][\@ne]{\relsize{+#1}#2}}
\textsmaller 676 \DeclareRobustCommand*\textsmaller[2][\@ne]{\relsize{-#1}#2}}
\largerr 677 \DeclareRobustCommand*\largerr{\relsize{+2}}
\smallerr 678 \DeclareRobustCommand*\smallerr{\relsize{-2}}

```

## \firstofone and the Queer \catcodes

Remember that once a macro's argument has been read, its \catcodes are assigned forever and ever. That's what is \firstofone for. It allows you to change the \catcodes locally for a definition *outside* the changed \catcodes' group. Just see the below usage of this macro 'with T<sub>E</sub>X's eyes', as my T<sub>E</sub>X Guru taught me.

```

689 \long\def\firstofone#1{#1}

The next command, \foone, is intended as two-argument for shortening of the
\bggroup...\firstofone{\egroup...} hack.

\foone 694 \long\def\foone#1{\bggroup#1\egroupfirstofone}
        696 \long\def\egroupfirstofone#1{\egroup#1}

\foeatletter 698 \long\def\foeatletter{\foone\makeatletter}

And this one is defined, I know, but it's not \long with the standard definition.

\gobble 705 \long\def\gobble#1{}
        706 \let\@gobble\gobble
\gobbletwo 707 \let\gobbletwo\@gobbletwo

```

## Some 'other' stuff

Here I define a couple of macros expanding to special chars made 'other'. It's important the cs are expandable and therefore they can occur e.g. inside \csname...\endcsname unlike e.g. cs'es \chardefed.

```

717 \foone{\catcode`\_ =8_\}%
\subs 718 {\let\subs=_}

720 \foone{\@makeother\_}%
\yiiunder 721 {\def\yiiunder{_\}}

723 \ifdefined\XeTeXversion

```

```

\xiiunder 724 \def\xiiunder{\char"005F\char"005F}%
725 \let\_ \xiiunder
726 \fi
728 \foone{\catcode`\ [=1\@makeother\{
729 \catcode`\ ]=2\@makeother\}}%
730 [%
\xiilbrace 731 \def\xiilbrace[{]%
\xiirbrace 732 \def\xiirbrace[}%
733 ]% of \firstofone

```

Note that L<sup>A</sup>T<sub>E</sub>X's `\@charlb` and `\@charrb` are of catcode 11 ('letter'), cf. The L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> Source file k, lines 129–130.

Now, let's define such a smart `_` (underscore) which will be usual `_8` in the math mode and `_12` ('other') outside math.

```

744 \foone{\catcode`\_=\active}
745 {%
\smartunder 746 \newcommand*\smartunder{%
747 \catcode`\_=\active
748 \def_{\ifmmode\subs\else\_ \fi}}}% We define it as \_ not just as \xiiunder
because some font encodings don't have _ at the \char`\_ position.
754 \foone{\catcode`\ !=o
755 \@makeother\ }
\xiibackslash 756 {\!newcommand*\xiibackslash{\}}
\bslash 760 \let\bslash=\xiibackslash
764 \foone{\@makeother\}%
\xiipercent 765 {\def\xiipercent{}}%
768 \foone{\@makeother\&}%
\xiiand 769 {\def\xiiand{&}}%
771 \foone{\@makeother\ }%
\xiispace 772 {\def\xiispace{\ }}%

```

We introduce `\visibleospace` from Will Robertson's `xltxtra` if available. It's not sufficient `\@ifpackageloaded{xltxtra}` since `\xxt@visibleospace` is defined only unless `no-verb` option is set. 2008/08/06 I recognized the difference between `\xiispace` which has to be plain 'other' char (used in `\xiistring`) and something visible to be printed in any font.

```

781 \AtBeginDocument{%
782 \ifdefined\xxt@visibleospace
783 \let\visibleospace\xxt@visibleospace
784 \else
785 \let\visibleospace\xiispace
786 \fi}

```

## Metasymbols

I fancy also another Knuthian trick for typesetting *metasymbols* in *The T<sub>E</sub>Xbook*. So I repeat it here. The inner `\meta` macro is copied verbatim from doc's v2.1b documentation dated 2004/02/09 because it's so beautifully crafted I couldn't resist. I only don't make it `\long`.

“The new implementation fixes this problem by defining `\meta` in a radically different way: we prevent hyphenation by defining a `\language` which has no patterns associated with it and use this to typeset the words within the angle brackets.”

```
\meta 807 \DeclareRobustCommand*\meta[1]{%
```

“Since the old implementation of `\meta` could be used in math we better ensure that this is possible with the new one as well. So we use `\ensuremath` around `\langle` and `\rangle`. However this is not enough: if `\meta@font@select` below expands to `\itshape` it will fail if used in math mode. For this reason we hide the whole thing inside an `\nfss@text` box in that case.”

```
815 \ensuremath\langle
816 \ifmmode\@xa\nfss@text\fi
817 {%
818 \meta@font@select
```

Need to keep track of what we changed just in case the user changes font inside the argument so we store the font explicitly.

```
826 #1\%
828 }\ensuremath\rangle
829 }
```

But I define `\meta@font@select` as the brutal and explicit `\it` instead of the original `\itshape` to make it usable e.g. in the `gmdoc`’s `\cs` macro’s argument.

```
\meta@font@select 837 \def\meta@font@select{\it}
```

The below `\meta`’s drag<sup>1</sup> is a version of *The T<sub>E</sub>Xbook*’s one.

```
\<...> 843 \def\<#1>{\meta{#1}}
```

## Macros for Printing Macros and Filenames

First let’s define three auxiliary macros analogous to `\dywiz` from `polski.sty`: a short-hands for `\discretionary` that’ll stick to the word not spoiling its hyphenability and that’ll won’t allow a linebreak just before nor just after themselves. The `\discretionary` T<sub>E</sub>X primitive has three arguments: #1 ‘before break’, #2 ‘after break’, #3 ‘without break’, remember?

```
\discre 854 \def\discre#1#2#3{\leavevmode\kernosp%
855 \discretionary{#1}{#2}{#3}\penalty10000\hskiposp\relax}
\discret 856 \def\discret#1{\leavevmode\kernosp%
857 \discretionary{#1}{#1}{#1}\penalty10000\hskiposp\relax}
```

A tiny little macro that acts like `\-` outside the math mode and has its original meaning inside math.

```
861 \def\:{\ifmmode\afterfi{\mskip\medmuskip}\else\afterfi{\discret{%
}\fi}
\vs 864 \newcommand*\{vs}{\discre{\visiblespace}{\visiblespace}}
```

Then we define a macro that makes the spaces visible even if used in an argument (i.e., in a situation where `re\catcodeing` has no effect).

```
\printspaces 871 \def\printspaces#1{{\let~\vs\let\ =\vs\gm@pswords#1\@nil}}
\gm@pswords 873 \def\gm@pswords#1#2\@nil{%
```

<sup>1</sup> Think of the drags that transform a very nice but rather standard ‘auntie’ (‘Tante’ in Deutsch) into a most adorable Queen ;-).

```

874 \ifx\relax#1\relax\else#1\fi
875 \ifx\relax#2\relax\else\vs\penalty\hyphenpenalty\gm@pswords#2\@@nil%
    \fi}% note that in the recursive call of \gm@pswords the argument string is
    not extended with a guardian space: it has been already by \printspaces.

```

```

\sfname 881 \DeclareRobustCommand*\sfname[1]{\textsf{\printspaces{#1}}}

```

```

\gm@discretionaryslash 883 \def\gm@discretionaryslash{\discre{/}{\hbox{}}{}}}% the second pseudo-
    argument nonempty to get \hyphenpenalty not \exhyphenpenalty.

```

```

\file 888 \DeclareRobustCommand*\file[1]{\gm@printslashes#1/%
    \gm@printslashes}

```

```

\gm@printslashes 890 \def\gm@printslashes#1/#2\gm@printslashes{%
891     \sfname{#1}%
892     \ifx\gm@printslashes#2\gm@printslashes
893     \else
894     \textsf{\gm@discretionaryslash}%
895     \afterfi{\gm@printslashes#2\gm@printslashes}\fi}

```

it allows the spaces in the filenames (and prints them as `\`).

The below macro I use to format the packages' names.

```

\pk 903 \DeclareRobustCommand*\pk[1]{\textsf{\textup{#1}}}

```

Some (if not all) of the below macros are copied from doc and/or ltxdoc.

A macro for printing control sequences in arguments of a macro. Robust to avoid writing an explicit `\` into a file. It calls `\ttfamily` not `\tt` to be usable in headings which are boldface sometimes.

```

\cs 914 \DeclareRobustCommand*\cs[2][\backslash]{%
\ 915     \def\--{\discretionary{\rmfamily-}{\relax}{\relax}}}%
916     \def{\char`\}\def\{\char`\}\ttfamily\char`#1#2}}

```

```

\env 920 \DeclareRobustCommand*\env[1]{\cs[] {#1}}

```

And for the special sequences like `^~A`:

```

923 \foone{\@makeother\^}
\hathat 924 {\DeclareRobustCommand*\hathat[1]{\cs[^^]{#1}}}

```

And one for encouraging linebreaks e.g., before long verbatim words.

```

\possfil 929 \newcommand*\possfil{\hfil\penalty1000\hfilneg}

```

The five macros below are taken from the ltxdoc.dtx.

`"\cmd{\foo}` Prints `\foo` verbatim. It may be used inside moving arguments. `\cs{foo}` also prints `\foo`, for those who prefer that syntax. (This second form may even be used when `\foo` is `\outer`)."

```

\cmd 939 \def\cmd#1{\cs{\@xa\cmd@to@cs\string#1}}
\cmd@to@cs 941 \def\cmd@to@cs#1#2{\char\number`#2\relax}
    \marg{text} prints {\text}, 'mandatory argument'.

```

```

\marg 945 \def\marg#1{\ttfamily\char`\}\meta{#1}{\ttfamily\char`\}}
    \oarg{text} prints [{text}], 'optional argument'. Also \oarg[text] does that.

```

```

\oarg 950 \def\oarg{\@ifnextchar[\@oargsq\@oarg}
\@oarg 952 \def\@oarg#1{\ttfamily\}\meta{#1}{\ttfamily}}
\@oargsq 953 \def\@oargsq[#1]{\@oarg{#1}}

```

`\parg{te,xt}` prints `(\te,xt)`, 'picture mode argument'.

```

\parg 957 \def\parg{\@ifnextchar(\@pargp\@parg}

```

```

\@parg 959 \def\@parg#1{\tfamily{}\meta{#1}\tfamily)}}
\@pargp 960 \def\@pargp(#1){\@parg{#1}}

But we can have all three in one command.

964 \AtBeginDocument{%
\arg 965 \let\math@arg\arg
\arg 966 \def\arg{\ifmmode\math@arg\else\afterfi{%
967 \ifnextchar[%
968 \@oargsq{\@ifnextchar(%
969 \@pargp\marg}}\fi}%
970 }

```

## Storing and Restoring the Meanings of CSs

First a Boolean switch of globalness of assignments and its verifier.

```

\ifgmu@SMglobal 976 \newif\ifgmu@SMglobal
\SMglobal 978 \def\SMglobal{\gmu@SMglobaltrue}

```

The subsequent commands are defined in such a way that you can ‘prefix’ them with `\SMglobal` to get global (re)storing.

A command to store the current meaning of a CS in another macro to temporarily redefine the CS and be able to set its original meaning back (when grouping is not recommended):

```

\StoreMacro 989 \def\StoreMacro{%
990 \bgroup\makeatletter\@ifstar\egStore@MacroSt\egStore@Macro}

```

The unstarred version takes a cs and the starred version a text, which is intended for special control sequences. For storing environments there is a special command in line 1113.

```

\egStore@Macro 995 \long\def\egStore@Macro#1{\egroup\Store@Macro{#1}}
\egStore@MacroSt 996 \long\def\egStore@MacroSt#1{\egroup\Store@MacroSt{#1}}

\Store@Macro 998 \long\def\Store@Macro#1{%
999 \escapechar92
1000 \ifgmu@SMglobal\afterfi\global\fi
1001 \@xa\let\csname_/gmu/store/string#1\endcsname#1%
1002 \global\gmu@SMglobalfalse}

\Store@MacroSt 1005 \long\def\Store@MacroSt#1{%
1006 \edef\gmu@smtempa{%
1007 \ifgmu@SMglobal\global\fi
1008 \@nx\let\@xa\@nx\csname/gmu/store/bslash#1\endcsname% we add back-
slash because to ensure compatibility between \ (Re)StoreMacro and
\ (Re)StoreMacro*, that is. to allow writing e.g. \StoreMacro\kitten
and then \RestoreMacro*{kitten} to restore the meaning of \kitten.
1013 \@xa\@nx\csname#1\endcsname}
1014 \gmu@smtempa
1015 \global\gmu@SMglobalfalse}% we wish the globality to be just once.

```

We make the `\StoreMacro` command a three-step to allow usage of the most inner macro also in the next command.

The starred version, `\StoreMacro*` works with csnames (without the backslash). It’s first used to store the meanings of robust commands, when you may need to store not only `\foo`, but also `\csname foo \endcsname`.

The next command iterates over a list of CSs and stores each of them. The CS may be separated with commas but they don't have to.

```

\StoreMacros 1031 \long\def\StoreMacros{\bgroup\makeatletter\Store@Macros}
\Store@Macros 1032 \long\def\Store@Macros#1{\egroup
1033   \gmu@setsetSMglobal
1034   \let\gml@StoreCS\Store@Macro
1035   \gml@storemacros#1.}

\gmu@setsetSMglobal 1038 \def\gmu@setsetSMglobal{%
1039   \ifgmu@SMglobal
1040     \let\gmu@setSMglobal\gmu@SMglobaltrue
1041   \else
1042     \let\gmu@setSMglobal\gmu@SMglobalfalse
1043   \fi}

```

And the inner iterating macro:

```

\gml@storemacros 1046 \long\def\gml@storemacros#1{%
\gmu@reserveda 1047   \def\gmu@reserveda{\@nx#1}% My TEX Guru's trick to deal with \fi and such,
                  i.e., to hide #1 from TEX when it is processing a test's branch without expand-
                  ing.
1050   \if\gmu@reserveda.% a dot finishes storing.
1051     \global\gmu@SMglobalfalse
1052   \else
1053     \if\gmu@reserveda,% The list this macro is put before may contain commas
                  and that's O.K., we just continue the work.
1055     \afterfifi\gml@storemacros
1056   \else% what is else this shall be stored.
1057     \gml@StoreCS{#1}% we use a particular CS to may \let it both to the storing
                  macro as above and to the restoring one as below.
1060     \afterfifi{\gmu@setSMglobal\gml@storemacros}%
1061     \fi
1062   \fi}

```

And for the restoring

```

\RestoreMacro 1069 \def\RestoreMacro{%
1070   \bgroup\makeatletter\@ifstar\egRestore@MacroSt\egRestore@Macro}

\egRestore@Macro 1072 \long\def\egRestore@Macro#1{\egroup\Restore@Macro{#1}}
\egRestore@MacroSt 1073 \long\def\egRestore@MacroSt#1{\egroup\Restore@MacroSt{#1}}

\Restore@Macro 1075 \long\def\Restore@Macro#1{%
1076   \escapechar92
1077   \ifgmu@SMglobal\afterfi\global\fi
1078   \@xa\let\@xa#1\cname_/gmu/store/string#1\endcsname
1079   \global\gmu@SMglobalfalse}

\Restore@MacroSt 1081 \long\def\Restore@MacroSt#1{%
1082   \edef\gmu@smtempa{%
1083     \ifgmu@SMglobal\global\fi
1084     \@nx\let\@xa\@nx\cname#1\endcsname
1085     \@xa\@nx\cname/gmu/store/bslash#1\endcsname}% cf. the commentary
                  in line 1008.
1087   \gmu@smtempa
1088   \global\gmu@SMglobalfalse}

\RestoreMacros 1091 \long\def\RestoreMacros{\bgroup\makeatletter\Restore@Macros}

```

```

\Restore@Macros 1093 \long\def\Restore@Macros#1{\egroup
1094 \gmu@setsetSMglobal
1095 \let\gml@storeCS\Restore@Macro% we direct the core CS towards restoring
and call the same iterating macro as in line 1035.
1098 \gml@storemacros#1.}

```

As you see, the `\RestoreMacros` command uses the same iterating macro inside, it only changes the meaning of the core macro.

And to restore *and* use immediately:

```

\StoredMacro 1104 \def\StoredMacro{\bgroup\makeatletter\Stored@Macro}
\Stored@Macro 1105 \long\def\Stored@Macro#1{\egroup\Restore@Macro#1#1}

```

To be able to call a stored cs without restoring it.

```

\storedcsname 1108 \def\storedcsname#1{%
1109 \csname_/gmu/store\bslash#1\endcsname}
2008/08/03 we need to store also an environment.

```

```

\StoreEnvironment 1113 \def\StoreEnvironment#1{%
1115 \StoreMacro*{#1}\StoreMacro*{end#1}}

```

```

\RestoreEnvironment 1117 \def\RestoreEnvironment#1{%
1119 \RestoreMacro*{#1}\RestoreMacro*{end#1}}

```

It happened (see the definition of `\@docinclude` in `gmdoc.sty`) that I needed to `\relax` a bunch of macros and restore them after some time. Because the macros were rather numerous and I wanted the code more readable, I wanted to `\do` them. After a proper defining of `\do` of course. So here is this proper definition of `\do`, provided as a macro (a declaration).

```

\StoringAndRelaxingDo 1134 \long\def\StoringAndRelaxingDo{%
1135 \gmu@SMdo@setscope
1136 \long\def\do##1{%
1137 \gmu@SMdo@scope
1138 \@xa\let\csname_/gmu/store/string##1\endcsname##1%
1139 \gmu@SMdo@scope\let##1\relax}}
\gmu@SMdo@setscope 1141 \def\gmu@SMdo@setscope{%
1142 \ifgmu@SMglobal\let\gmu@SMdo@scope\global
1143 \else\let\gmu@SMdo@scope\relax
1144 \fi
1145 \global\gmu@SMglobalfalse}

```

And here is the counter-definition for restore.

```

\RestoringDo 1154 \long\def\RestoringDo{%
1155 \gmu@SMdo@setscope
1156 \long\def\do##1{%
1157 \gmu@SMdo@scope
1158 \@xa\let\@xa##1\csname_/gmu/store/string##1\endcsname}}

```

Note that both `\StoringAndRelaxingDo` and `\RestoringDo` are sensitive to the `\SMglobal` ‘prefix’.

And to store a cs as explicitly named cs, i.e. to `\let` one csname another (`\n@melet` not `\@namelet` because the latter is defined in Till Tantau’s beamer class another way) (both arguments should be text):

```

\n@melet 1167 \def\n@melet#1#2{%
1168 \edef\gmu@nl@reserveda{%

```



```

1169 \let\@xa\@nx\csname#1\endcsname
1170 \@xa\@nx\csname#2\endcsname}%
1171 \gmu@nl@reserveda}

```

The \global prefix doesn't work with \n@melet so we define the alternative.

```

\gn@melet 1175 \def\gn@melet#1#2{%
1176 \edef\gmu@nl@reserveda{%
1177 \global\let\@xa\@nx\csname#1\endcsname
1178 \@xa\@nx\csname#2\endcsname}%
1179 \gmu@nl@reserveda}

```

## Not only preamble!

Let's remove some commands from the list to erase at begin document! Primarily that list was intended to save memory not to forbid anything. Nowadays, when memory is cheap, the list of only-preamble commands should be rethought IMO.

```

\not@onlypreamble 1196 \newcommand\not@onlypreamble[1]{%
1197 \def\do##1{\ifx#1##1\else\@nx\do\@nx##1\fi}%
1198 \xdef\@preamblecmds{\@preamblecmds}}
1200 \not@onlypreamble\@preamblecmds
1201 \not@onlypreamble\ifpackageloaded
1202 \not@onlypreamble\ifclassloaded
1203 \not@onlypreamble\ifl@aded
1204 \not@onlypreamble\@pkgextension

```

And let's make the message of only preamble command's forbidden use informative a bit:

```

\gm@notprerr 1209 \def\gm@notprerr{\can_be_used_only_in_preamble(\on@line)}
1211 \AtBeginDocument{%
1212 \def\do#1{\@nx\do\@nx#1}%
1213 \edef\@preamblecmds{%
1214 \def\@nx\do##1{%
1215 \def##1{\@nx\PackageError{gmutils/LaTeX}%
1216 {\@nx\string##1\@nx\gm@notprerr}\@nx\@eha}}%
1217 \@preamblecmds}}

```

A subtle error raises: the L<sup>A</sup>T<sub>E</sub>X standard \@onlypreamble and what \document does with \@preamblecmds makes any two of 'only preamble' cs's \ifx-identical inside document. And my change makes any two cs's \ifx-different. The first it causes a problem is \nocite that checks \ifx\@onlypreamble\document. So hoping this is a rare problem, we circumvent in with

```

\nocite 1227 \def\nocite#1{%
1228 \@bsphack{\setboxo=\hbox{\cite{#1}}}\@esphack}

```

## Third Person Pronouns

Is a reader of my documentations 'she' or 'he' and does it make a difference?

Not to favour any gender in the personal pronouns, define commands that'll print alternately masculine and feminine pronoun of third person. By 'any' I mean not only typically masculine and typically feminine but the entire amazingly rich variety of people's genders, *including* those who do not describe themselves as 'man' or 'woman'.

One may say two pronouns is far too little to cover this variety but I could point Ursula's K. LeGuin's *The Left Hand Of Darkness* as another acceptable answer. In that moody and moderate SF novel the androgynous persons are usually referred to as 'mister', 'sir' or 'he': the meaning of reference is extended. Such an extension also my automatic pronouns do suggest. It's *not* political correctness, it's just respect to people's diversity.

```
gm@PronounGender 1257 \newcounter{gm@PronounGender}
\gm@atppron 1259 \newcommand*\gm@atppron[2]{%
1260 \stepcounter{gm@PronounGender}% remember \stepcounter is global.
1261 \ifodd\value{gm@PronounGender}#1\else#2\fi}

\heshe 1263 \newcommand*\heshe{\gm@atppron{he}{she}}
\hisher 1264 \newcommand*\hisher{\gm@atppron{his}{her}}
\himher 1265 \newcommand*\himher{\gm@atppron{him}{her}}
\hishers 1266 \newcommand*\hishers{\gm@atppron{his}{hers}}

\HeShe 1268 \newcommand*\HeShe{\gm@atppron{He}{She}}
\HisHer 1269 \newcommand*\HisHer{\gm@atppron{His}{Her}}
\HimHer 1270 \newcommand*\HimHer{\gm@atppron{Him}{Her}}
\HisHers 1271 \newcommand*\HisHers{\gm@atppron{His}{Hers}}
```

## To Save Precious Count Registers

It's a contribution to T<sub>E</sub>X's ecology ;-). You can use as many CSs as you wish and you may use only 256 count registers (although in  $\varepsilon$ -T<sub>E</sub>X there are 2<sup>16</sup> count registers, which makes the following a bit obsolete).

```
\nummacro 1280 \newcommand*\nummacro[1]{\gdef#1{o}}
\stepnummacro 1282 \newcommand*\stepnummacro[1]{%
1283 \@tempcnta=#1\relax
1284 \advance\@tempcnta_by1\relax
1285 \xdef#1{\the\@tempcnta}}% Because of some mysterious reasons explicit \counto
interferred with page numbering when used in \gmd@evpaddonce in gm-
doc.

\addtonummacro 1291 \newcommand*\addtonummacro[2]{%
1292 \counto=#1\relax
1293 \advance\countoby#2\relax
1294 \xdef#1{\the\count\z@}}
```

Need an explanation? The `\nummacro` declaration defines its argument (that should be a CS) as `{o}` which is analogous to `\newcount` declaration but doesn't use up any count register.

Then you may use this numeric macro as something between T<sub>E</sub>X's count CS and L<sup>A</sup>T<sub>E</sub>X's counter. The macros `\stepnummacro` and `\addtonummacro` are analogous to L<sup>A</sup>T<sub>E</sub>X's `\stepcounter` and `\addtocounter` respectively: `\stepnummacro` advances the number stored in its argument by 1 and `\addtonummacro` advances it by the second argument. As the L<sup>A</sup>T<sub>E</sub>X's analogoi, they have the global effect (the effect of global warming ;-)).

So far I've used only `\nummacro` and `\stepnummacro`. Notify me if you use them and whether you need sth. more, `\multiplynummacro` e.g.

## Improvements to mwcls Sectioning Commands

That is, ‘Expe-ri-mente’<sup>2</sup> mit MW sectioning & \refstepcounter to improve mwcls’s cooperation with hyperref. They shouldn’t make any harm if another class (non-mwcls) is loaded.

We \refstep sectioning counters even if the sectionings are not numbered, because otherwise

1. pdfTeX cried of multiply defined \labels,
2. e.g. in a table of contents the hyperlink <rozdzia\l\ Kwiaty polskie> linked not to the chapter’s heading but to the last-before-it change of \ref.

1329 \AtBeginDocument{% because we don’t know when exactly hyperref is loaded and maybe after this package.

NoNumSecs 1331 \@ifpackageloaded{hyperref}{\newcounter{NoNumSecs}%  
1332 \setcounter{NoNumSecs}{617}% to make \refing to an unnumbered section visible (and funny?).

\gm@hyperrefstepcounter 1334 \def\gm@hyperrefstepcounter{\refstepcounter{NoNumSecs}}%  
\gm@targetheading 1335 \DeclareRobustCommand\*\gm@targetheading[1]{%  
1336 \hypertarget{#1}{#1}}}% end of then  
\gm@hyperrefstepcounter 1337 {\def\gm@hyperrefstepcounter{}%  
\gm@targetheading 1338 \def\gm@targetheading#1{#1}}}% end of else  
1339 }% of \AtBeginDocument

Auxiliary macros for the kernel sectioning macro:

bersectionsoutofmainmatter 1342 \def\gm@dontnumbersectionsoutofmainmatter{%  
1343 \if@mainmatter\else\HeadingNumberedfalse\fi}  
gm@clearpagesduetoopenright 1344 \def\gm@clearpagesduetoopenright{%  
1345 \if@openright\cleardoublepage\else\clearpage\fi}

To avoid \defing of \mw@sectionxx if it’s undefined, we redefine \def to gobble the definition and restore the original meaning of itself.

Why shouldn’t we change the ontological status of \mw@sectionxx (not define if undefined)? Because some macros (in gmdocc e.g.) check it to learn whether they are in an mwcls or not.

But let’s make a shorthand for this test since we’ll use it three times in this package and maybe also somewhere else.

\@ifnotmw 1358 \long\def\@ifnotmw#1#2{\@ifundefined{mw@sectionxx}{#1}{#2}}  
1360 \let\gmu@def\def  
\@ifnotmw 1361 \@ifnotmw{%  
\gmu@def 1362 \StoreMacro\gmu@def\def\gmu@def#1#2{\RestoreMacro\gmu@def}}{-}

I know it may be of bad taste (to write such a way *here*) but I feel so lonely and am in an alien state of mind after 3 hour sleep last night and, worst of all, listening to sir Edward Elgar’s flamboyant Symphonies d’Art Nouveau.

A decent person would just wrap the following definition in \@ifundefined’s Else. But look, the definition is so long and I feel so lonely etc. So, I define \def (for some people there’s nothing sacred) to be a macro with two parameters, first of which is delimited by digit 4 (the last token of \mw@sectionxx’s parameter string) and the latter is undelimited which means it’ll be the body of the definition. Such defined \def does nothing else but restores its primitive meaning by the way sending its arguments to the Gobbled Tokens’ Paradise. Luckily, \RestoreMacro contains \let not \def.

The kernel of MW’s sectioning commands:

<sup>2</sup> A. Berg, Wozzeck.

```

1381 \gmu@def\mw@sectionxx#1#2[#3]#4{%
1382   \edef\mw@HeadingLevel{\csname_#1@level\endcsname
1383     \space}% space delimits level number!
1384   \ifHeadingNumbered
1385     \ifnum_#1\mw@HeadingLevel>\c@secnumdepth_#1%
       \HeadingNumberedfalse_#1\fi

line below is in ifundefined to make it work in classes other than mwbk

1388     \@ifundefined{if@mainmatter}{_#1}{%
       \gm@dontnumbersectionsoutofmainmatter}
1389   \fi

%   \ifHeadingNumbered
%   \refstepcounter{#1}%
%   \protected@edef\HeadingNumber{\csname
%     the#1\endcsname\relax}%
%   \else
%   \let\HeadingNumber\@empty
%   \fi

\HeadingRHeadText 1398 \def\HeadingRHeadText{#2}%
\HeadingTOCText    1399 \def\HeadingTOCText{#3}%
\HeadingText       1400 \def\HeadingText{#4}%
\mw@HeadingType    1401 \def\mw@HeadingType{#1}%
1402 \if\mw@HeadingBreakBefore
1403   \if@specialpage\else\thispagestyle{closing}\fi
1404   \@ifundefined{if@openright}{_#1}{\gm@clearpagesduetoopenright}%
1405   \if\mw@HeadingBreakAfter
1406     \thispagestyle{blank}\else
1407     \thispagestyle{opening}\fi
1408     \global\@topnum\z@
1409   \fi% of \if\mw@HeadingBreakBefore

placement of \refstep suggested by me (GM)

1412   \ifHeadingNumbered
1413     \refstepcounter{#1}%
1414     \protected@edef\HeadingNumber{\csname_#1\endcsname\relax}%
1415   \else
1416     \let\HeadingNumber\@empty
1417     \gm@hyperrefstepcounter
1418   \fi% of \ifHeadingNumbered

1420 \if\mw@HeadingRunIn
1421   \mw@runinheading
1422 \else
1423   \if\mw@HeadingWholeWidth
1424     \if@twocolumn
1425       \if\mw@HeadingBreakAfter
1426       \onecolumn
1427       \mw@normalheading
1428       \pagebreak\relax
1429       \if@twoside
1430       \null
1431       \thispagestyle{blank}%
1432       \newpage

```

```

1433         \fi% of \if@twoside
1434     \twocolumn
1435     \else
1436         \@topnewpage[\mw@normalheading]%
1437         \fi% of \if\mw@HeadingBreakAfter
1438     \else
1439         \mw@normalheading
1440         \if\mw@HeadingBreakAfter\pagebreak\relax\fi
1441     \fi% of \if@twocolumn
1442 \else
1443     \mw@normalheading
1444     \if\mw@HeadingBreakAfter\pagebreak\relax\fi
1445 \fi% of \if\mw@HeadingWholeWidth
1446 \fi% of \if\mw@HeadingRunIn
1447 }

```

### An improvement of MW's \SetSectionFormatting

A version of MW's \SetSectionFormatting that lets to leave some settings unchanged by leaving the respective argument empty ({} or []).

Notice: If we adjust this command for new version of MWCLS, we should name it \SetSectionFormatting and add issuing errors if the inner macros are undefined.

- #1 (optional) the flags, e.g. breakbefore, breakafter;
- #2 the sectioning name, e.g. chapter, part;
- #3 preskip;
- #4 heading type;
- #5 postskip

```

1470 \relaxen\SetSectionFormatting
\SetSectionFormatting 1471 \newcommand*\SetSectionFormatting[5][\empty]{%
1472     \ifx\empty#1\relax\else% empty (not \empty!) #1 also launches \else.
\mw@HeadingRunIn 1473     \def\mw@HeadingRunIn{10}\def\mw@HeadingBreakBefore{10}%
\mw@HeadingBreakBefore 1474     \def\mw@HeadingBreakAfter{10}\def\mw@HeadingWholeWidth{10}%
\mw@HeadingBreakAfter 1475     \@ifempty{#1}{}{\mw@processflags#1,\relax}% If #1 is omitted, the flags
\mw@HeadingWholeWidth are left unchanged. If #1 is given, even as [], the flags are first cleared and
then processed again.
1478     \fi
1479     \@ifundefined{#2}{\@namedef{#2}{\mw@section{#2}}}{}%
1480     \mw@secdef{#2}{@preskip}{#3}{2\oblig.}%
1481     \mw@secdef{#2}{@head}{#4}{3\oblig.}%
1482     \mw@secdef{#2}{@postskip}{#5}{4\oblig.}%
1483     \ifx\empty#1\relax
1484         \mw@secundef{#2@flags}{1\optional)}%
1485     \else\mw@setflags{#2}%
1486     \fi}
\mw@secdef 1488 \def\mw@secdef#1#2#3#4{% #1 the heading name,
% #2 the command distinctior,
% #3 the meaning,
% #4 the number of argument to error message.
1492     \@ifempty{#3}
1493         {\mw@secundef{#1#2}{#4}}
1494         {\@namedef{#1#2}{#3}}
\mw@secundef 1496 \def\mw@secundef#1#2{%

```

```

1497 \@ifundefined{#1}{%
1498   \ClassError{mwcls/gm}{%
1499     command\backslash#1\undefined\MessageBreak
1500     after\backslashSetSectionFormatting!!!\MessageBreak}{%
1501     Provide the #2 argument of \backslash
        SetSectionFormatting.}}{}}

```

First argument is a sectioning command (wo. \) and second the stuff to be added at the beginning of the heading declarations.

```

\addtoheading 1506 \def\addtoheading#1#2{%
1507   \n@melet{gmu@reserveda}{#1@head}%
1508   \toks\z@=\@xa{gmu@reserveda}%
1509   \toks\tw@={#2}%
1510   \edef\gmu@reserveda{\the\toks\tw@\the\toks\z@}%
1511   \n@melet{#1@head}{gmu@reserveda}%
1513 }

```

### Negative \addvspace

When two sectioning commands appear one after another (we may assume that this occurs only when a lower section appears immediately after higher), we prefer to put the *smaller* vertical space not the larger, that is, the preskip of the lower sectioning not the postskip of the higher.

For that purpose we modify the very inner macros of MWCLS to introduce a check whether the previous vertical space equals the postskip of the section one level higher.

```

1525 \@ifnotmw{}{% We proceed only in MWCLS

```

The information that we are just after a heading will be stored in the \gmu@prevsec macro: any heading will define it as the section name and \everypar (any normal text) will clear it.

```

\@afterheading 1530 \def\@afterheading{%
1531   \@nobreaktrue
1532   \xdef\gmu@prevsec{\mw@HeadingType}% added now
1533   \everypar{%
1534     \grelaxen\gmu@prevsec% added now. All the rest is original LATEX.
1535     \if@nobreak
1536     \@nobreakfalse
1537     \clubpenalty\@M
1538     \if@afterindent\else
1539     {\setbox\z@\lastbox}%
1540     \fi
1541     \else
1542     \clubpenalty\@clubpenalty
1543     \everypar{}%
1544     \fi}}

```

If we are (with the current heading) just after another heading (one level lower I suppose), then we add the less of the higher header's post-skip and the lower header pre-skip or, if defined, the two-header-skip. (We put the macro defined below just before \addvspace in MWCLS inner macros.)

```

\gmu@checkaftersec 1551 \def\gmu@checkaftersec{%
1552   \@ifundefined{gmu@prevsec}{}%
1553   \ifgmu@postsec% an additional switch that is true by default but may be
        turned into an \ifdim in special cases, see line 1589.

```

```

1556 {\xa\mw@getflags\x\xa\gmu@prevsec}%
1557 \glet\gmu@reserveda\mw@HeadingBreakAfter}%
\gmu@reserveda 1558 \if\mw@HeadingBreakBefore\def\gmu@reserveda{11}\fi% if the current
    heading inserts page break before itself, all the play with vskips is irrele-
    vant.
1561 \if\gmu@reserveda\else
1562 \penalty10000\relax
1563 \skip\z@=\csname\gmu@prevsec_\@postskip\endcsname\relax
1564 \skip\tw@=\csname\mw@HeadingType_\@preskip\endcsname\relax
1565 \@ifundefined{\mw@HeadingType_\@twoheadskip}{
1566 \ifdim\skip\z@>\skip\tw@
1567 \vskip-\skip\z@% we strip off the post-skip of previous header if it's bigger
    than current pre-skip
1569 \else
1570 \vskip-\skip\tw@% we strip off the current pre-skip otherwise
1571 \fi}{% But if the two-header-skip is defined, we put it
1573 \penalty10000
1574 \vskip-\skip\z@
1575 \penalty10000
1576 \vskip-\skip\tw@
1577 \penalty10000
1578 \vskip\csname\mw@HeadingType_\@twoheadskip\endcsname
1579 \relax}%
1580 \penalty10000
1581 \hrule\height\z@\relax% to hide the last (un)skip before subsequent \addvspaces.
1583 \penalty10000
1584 \fi
1585 \fi
1586 }% of \@ifundefined{gmu@prevsec} 'else'
1587 }% of \def\gmu@checkaftersec

\ParanoidPostsec 1589 \def\ParanoidPostsec{% this version of \ifgmu@postsec is intended for the spe-
    cial case of sections may contain no normal text, as while gmdocing.
\ifgmu@postsec 1592 \def\ifgmu@postsec{% note this macro expands to an open \if.
1593 \skip\z@=\csname\gmu@prevsec_\@postskip\endcsname\relax
1594 \ifdim\lastskip=\skip\z@\relax% we play with the vskips only if the last
    skip is the previous heading's postskip (a counter-example I met while
    gmdocing).
1598 }}
1600 \let\ifgmu@postsec\iftrue

\gmu@getaddvs 1602 \def\gmu@getaddvs#1\addvspace#2\gmu@getaddvs{%
1603 \toks\z@={#1}
1604 \toks\tw@={#2}}

    And the modification of the inner macros at last:
\gmu@setheading 1607 \def\gmu@setheading#1{%
1608 \xa\gmu@getaddvs#1\gmu@getaddvs
1609 \edef#1{%
1610 \the\toks\z@\@nx\gmu@checkaftersec
1611 \@nx\addvspace\the\toks\tw@}}
1613 \gmu@setheading\mw@normalheading
1614 \gmu@setheading\mw@runinheading

```

```

\SetTwoheadSkip 1616 \def\SetTwoheadSkip#1#2{\@namedef{#1@twoheadskip}{#2}}
1618 }% of \@ifnotmw

```

### My heading setup for mwcls

The setup of heading skips was tested in ‘real’ typesetting, for money that is. The skips are designed for 11/13 pt leading and together with my version of mw11.clo option file for mwcls make the headings (except paragraph and subparagraph) consist of an integer number of lines. The name of the declaration comes from my employer, “Wiedza Powszechna” Editions.

```

\WPheadings 1630 \@ifnotmw{}{% We define this declaration only when in mwcls.
1631 \def\WPheadings{%
1632   \SetSectionFormatting[breakbefore,wholewidth]
1633     {part}{\z@\@plus1fill}{\z@\@plus3fill}%
1635   \@ifundefined{chapter}{}{%
1636     \SetSectionFormatting[breakbefore,wholewidth]
1637       {chapter}
1638       {66\p@}% {67\p@} for Adventor/Schola 0,95.
1639       {\FormatHangHeading{\LARGE}}
1640       {27\p@\@plus0,2\p@\@minus1\p@}%
1641   }%
1643   \SetTwoheadSkip{section}{27\p@\@plus0,5\p@}%
1644   \SetSectionFormatting{section}
1645     {24\p@\@plus0,5\p@\@minus5\p@}%
1646     {\FormatHangHeading{\Large}}
1647     {10\p@\@plus0,5\p@}% ed. Krajewska of “Wiedza Powszechna”, as we un-
        derstand her, wants the skip between a heading and text to be rigid.
1651   \SetTwoheadSkip{subsection}{11\p@\@plus0,5\p@\@minus1\p@}%
1652   \SetSectionFormatting{subsection}
1653     {19\p@\@plus0,4\p@\@minus6\p@}
1654     {\FormatHangHeading{\large}}% 12/14 pt
1655     {6\p@\@plus0,3\p@}% after-skip 6 pt due to p.12, not to squeeze the before-
        skip too much.
1658   \SetTwoheadSkip{subsubsection}{10\p@\@plus1,75\p@\@minus1\p@}%
1659   \SetSectionFormatting{subsubsection}
1660     {10\p@\@plus0,2\p@\@minus1\p@}
1661     {\FormatHangHeading{\normalsize}}
1662     {3\p@\@plus0,1\p@}% those little skips should be smaller than you calcu-
        late out of a geometric progression, because the interline skip enlarges
        them.
1666   \SetSectionFormatting[runin]{paragraph}
1667     {7\p@\@plus0,15\p@\@minus1\p@}
1668     {\FormatRunInHeading{\normalsize}}
1669     {2\p@}%
1671   \SetSectionFormatting[runin]{subparagraph}
1672     {4\p@\@plus1\p@\@minus0,5\p@}
1673     {\FormatRunInHeading{\normalsize}}
1674     {\z@}%
1675 }% of \WPheadings
1676 }% of \@ifnotmw

```



## Compatibilising Standard and mwcls Sectionings

If you use Marcin Woliński’s document classes (mwcls), you might have met their little queerness: the sectioning commands take two optional arguments instead of standard one. It’s reasonable since one may wish one text to be put into the running head, another to the toc and yet else to the page. But the order of optionalities causes an incompatibility with the standard classes: MW section’s first optional argument goes to the running head not to toc and if you’ve got a source file written with the standard classes in mind and use the first (and only) optional argument, the effect with mwcls would be different if not error.

Therefore I counter-assign the commands and arguments to reverse the order of optional arguments for sectioning commands when mwcls are in use and reverse, to make mwcls-like sectioning optionals usable in the standard classes.

With the following in force, you may both in the standard classes and in mwcls give a sectioning command one or two optional arguments (and mandatory the last, of course). If you give just one optional, it goes to the running head and to toc as in scls (which is unlike in mwcls). If you give two optionals, the first goes to the running head and the other to toc (like in mwcls and unlike in scls).

(In both cases the mandatory last argument goes only to the page.)

What more is unlike in scls, it’s that even with them the starred versions of sectioning commands allow optionals (but they still send them to the Gobbled Tokens’ Paradise).

(In mwcls, the only difference between starred and non-starred sec commands is (not) numbering the titles, both versions make a contents line and a mark and that’s not changed with my redefinitions.)

```
1717 \@ifnotmw{% we are not in mwcls and want to handle mwcls-like sectionings i.e.,
      those written with two optionals.
\gm@secini 1720 \def\gm@secini{gm@la}%
\gm@secxx 1722 \def\gm@secxx#1#2[#3]#4{%
1723 \ifx\gm@secstar\@empty
1724 \n@melet{gm@true@#1mark}{#1mark}% a little trick to allow a special ver-
      sion of the heading just to the running head.
1726 \@namedef{#1mark}##1{% we redefine \<sec>mark to gobble its argument
      and to launch the stored true marking command on the appropriate
      argument.
1729 \csname_\gm@true@#1mark\endcsname{#2}%
1730 \n@melet{#1mark}{gm@true@#1mark}% after we’ve done what we wanted
      we restore original \#1mark.
1732 }%
\gm@secstar 1733 \def\gm@secstar{[#3]}% if \gm@secstar is empty, which means the sec-
      tioning command was written starless, we pass the ‘true’ sectioning
      command #3 as the optional argument. Otherwise the sectioning com-
      mand was written with star so the ‘true’ s.c. takes no optional.
1738 \fi
1739 \@xa\@xa\csname\gm@secini#1\endcsname
1740 \gm@secstar{#4}}%
1742 }{% we are in mwcls and want to reverse MW’s optionals order i.e., if there’s just one
      optional, it should go both to toc and to running head.
\gm@secini 1745 \def\gm@secini{gm@mw}%
1747 \let\gm@secmarkh\@gobble% in mwcls there’s no need to make tricks for special
      version to running headings.
\gm@secxx 1750 \def\gm@secxx#1#2[#3]#4{%
1751 \@xa\@xa\csname\gm@secini#1\endcsname
```

```

1752     \gm@secstar[#2][#3]{#4}}%
1753 }
\gm@sec 1755 \def\gm@sec#1{\@dblarg{\gm@secx{#1}}}
\gm@secx 1756 \def\gm@secx#1[#2]{%
1757     \@ifnextchar[{\gm@secxx{#1}{#2}}{\gm@secxx{#1}{#2}[#2]}}% if there's
        only one optional, we double it not the mandatory argument.
\gm@straightensec 1761 \def\gm@straightensec#1{% the parameter is for the command's name.
1762     \@ifundefined{#1}{}{% we don't change the ontological status of the command
        because someone may test it.
1764     \n@melet{\gm@secini#1}{#1}%
1765     \@namedef{#1}{%
\gm@secstar 1766     \@ifstar{\def\gm@secstar{*}\gm@sec{#1}}{%
\gm@secstar 1767     \def\gm@secstar{}\gm@sec{#1}}}%
1768 }%
1770 \let\do\gm@straightensec
1771 \do{part}\do{chapter}\do{section}\do{subsection}\do{%
        subsubsection}
1772 \@ifnotmw{}{\do{paragraph}}% this 'straightening' of \paragraph with the stan-
        dard article caused the 'TEX capacity exceeded' error. Anyway, who on Earth
        wants paragraph titles in toc or running head?

```

## enumerate\* and itemize\*

We wish the starred version of enumerate to be just numbered paragraphs. But hyperref redefines \item so we should do it a smart way, to set the L<sup>A</sup>T<sub>E</sub>X's list parameters that is.

(Marcin Woliński in mwcls defines those environments slightly different: his item labels are indented, mine are not; his subsequent paragraphs of an item are not indented, mine are.)

```

enumerate* 1788 \@namedef{enumerate*}{%
1789     \ifnum\@enumdepth>\thr@@
1790     \@toodeep
1791     \else
1792     \advance\@enumdepth\@ne
1793     \edef\@enumctr{enum\romannumeral\the\@enumdepth}%
1794     \@xa\list\csname\label\@enumctr\endcsname{%
1795     \partopsep\topsep\topsep\z@\leftmargin\z@
1796     \itemindent\@parindent\advance\itemindent\labelsep
1797     \labelwidth\@parindent
1798     \advance\labelwidth-\labelsep
1799     \listparindent\@parindent
1800     \usecounter\@enumctr
1801     \def\makelabel##1{##1\hfil}}%
1802     \fi}
1803 \@namedef{endenumerate*}{\endlist}
itemize* 1806 \@namedef{itemize*}{%
1807     \ifnum\@itemdepth>\thr@@
1808     \@toodeep
1809     \else
1810     \advance\@itemdepth\@ne

```

```

1811 \edef\@itemitem{labelitem\romannumeral\the\@itemdepth}%
1812 \@xa\list\csname\@itemitem\endcsname{%
1813 \partopsep\topsep_\topsep\z@_\leftmargin\z@
1814 \itemindent\@parindent
1815 \labelwidth\@parindent
1816 \advance\labelwidth-\labelsep
1817 \listparindent\@parindent
1818 \def\makelabel##1{##1\hfil_}}%
1819 \fi}
1820 \@namedef{enditemize*}{\endlist}

```

## The Logos

We'll modify The L<sup>A</sup>T<sub>E</sub>X logo now to make it fit better to various fonts.

```

1829 \let\oldLaTeX\LaTeX
1830 \let\oldLaTeXe\LaTeXe
1832 \def\TeX{T\kern-.1667em\lower.5ex\hbox{E}\kern-.125emX\@}
\DeclareLogo 1834 \newcommand*\DeclareLogo[3][\relax]{%
    #1 is for non-LATEX spelling and will be used in the PD1 encoding (to make pdf book-
    marks);
    #2 is the command, its name will be the PD1 spelling by default,
    #3 is the definition for all the font encodings except PD1.
\gmu@reserveda 1840 \ifx\relax#1\def\gmu@reserveda{\@xa@gobble\string#2}%
1841 \else
\gmu@reserveda 1842 \def\gmu@reserveda{#1}%
1843 \fi
1844 \edef\gmu@reserveda{%
\@nx 1845 \@nx\DeclareTextCommand\@nx#2{PD1}{\gmu@reserveda}}
1846 \gmu@reserveda
1847 \DeclareTextCommandDefault#2{#3}%
\DeclareRobustCommand* 1848 \DeclareRobustCommand*#2{#3}}% added for XYLATEX
\DeclareLogo 1851 \DeclareLogo\LaTeX{%
1852 {%
1853 L%
1854 \setbox\z@\hbox{\check@mathfonts
1855 \fontsize\sf@size\z@
1856 \math@fontsfalse\selectfont
1857 A}%
1858 \kern-.57\wd\z@
1859 \sbox\tw@_T%
1860 \vbox_to\ht\tw@{\copy\z@_vss}%
1861 \kern-.2\wd\z@}% originally -, 15em for T.
1862 {%
1863 \ifdim\fontdimen1\font=\z@
1864 \else
1865 \count\z@=\fontdimen5\font
1866 \multiply\count\z@_by_64\relax
1867 \divide\count\z@_by\p@
1868 \count\tw@=\fontdimen1\font
1869 \multiply\count\tw@_by\count\z@
1870

```

```

1871      \divide\count\tw@_by_64\relax
1872      \divide\count\tw@_by\tw@
1873      \kern-\the\count\tw@_sp\relax
1874      \fi}%
1875      \TeX}

\LaTeXe 1877 \DeclareLogo\LaTeXe{\mbox{\m@th_ \if
1878      b\expandafter\@car\f@series\@nil\boldmath\fi
1879      \LaTeX\kern.15em2$_{\textstyle\varepsilon}$}}

1881 \StoreMacro\LaTeX
1882 \StoreMacro*\LaTeX_

‘(L)TeX’ in my opinion better describes what I work with/in than just ‘LTeX’.

\LaTeXpar 1888 \DeclareLogo[(La)TeX]{\LaTeXpar}{%
1889      {%
1890      \setbox\z@\hbox{({}% )
1891      \copy\z@
1892      \kern-.2\wd\z@_L%
1893      \setbox\z@\hbox{\check@mathfonts
1894      \fontsize\sf@size\z@
1895      \math@fontsfalse\selectfont
1896      A}%
1897      \kern-.57\wd\z@
1898      \sbox\tw@_T%
1899      \vbox_ to\ht\tw@{\box\z@%
1900      \vss}%
1901      }%
1902      \kern-.07em% originally -,15 em for T.
1903      {%(
1904      \sbox\z@)%
1905      \kern-.2\wd\z@\copy\z@
1906      \kern-.2\wd\z@}\TeX
1907      }

```

“Here are a few definitions which can usefully be employed when documenting package files: now we can readily refer to  $\mathcal{M}\mathcal{S}$ -TeX, BibTeX and S<sub>L</sub>TeX, as well as the usual TeX and LTeX. There’s even a PLAIN TeX and a WEB.”

```

1914 \@ifundefined{AmSTeX}
\AmSTeX 1915 {\def\AmSTeX{\leavevmode\hbox{$\mathcal_A\kern-.2em%
1916      \lower.376ex%
1917      \hbox{$\mathcal_M$}\kern-.2em\mathcal_S$-\TeX}}}{%
1918 \DeclareLogo\BibTeX{{\rmfamily_B\kern-.05em%
1919      \textsc{i{\kern-.025em}b}\kern-.08em% the kern is wrapped in braces
1920      for my \fakescups’ sake.
1921      \TeX}}

\SLiTeX 1924 \DeclareLogo\SLiTeX{{\rmfamily_S\kern-.06emL\kern-.18em%
1925      \raise.32ex\hbox
1926      {\scshape_i}\kern_-.03em\TeX}}

\PlainTeX 1927 \DeclareLogo\PlainTeX{\textsc{Plain}\kern2pt\TeX}

\Web 1929 \DeclareLogo\Web{\textsc{Web}}

```

There’s also the (L)TeX logo got with the \LaTeXpar macro provided by gmutils. And here *The T<sub>E</sub>Xbook*’s logo:

```

\TeXbook 1932 \DeclareLogo[The_\TeX_book]\TeXbook{\textsl{The_\TeX_book}}
1933 \let\TB\TeXbook% TUG Boat uses this.

\TeX 1935 \DeclareLogo[e-TeX]\eTeX{%
1936 \ensuremath{\varepsilon}-\kern-.125em\TeX}% definition sent by Karl Berry
from TUG Boat itself.

\pdfTeX 1939 \DeclareLogo[pdfe-TeX]\pdfTeX{pdf\eTeX}

\pdfTeX 1941 \DeclareLogo\pdfTeX{pdf\TeX}

1943 \@ifundefined{XeTeX}{%
\XeTeX 1944 \DeclareLogo\XeTeX{X\kern-.125em\relax
1945 \@ifundefined{reflectbox}{%
1946 \lower.5ex\hbox{E}\kern-.1667em\relax}{%
1947 \lower.5ex\hbox{\reflectbox{E}}\kern-.1667em\relax}%
1948 \TeX}}{}

1950 \@ifundefined{XeLaTeX}{%
\XeLaTeX 1951 \DeclareLogo\XeLaTeX{X\kern-.125em\relax
1952 \@ifundefined{reflectbox}{%
1953 \lower.5ex\hbox{E}\kern-.1667em\relax}{%
1954 \lower.5ex\hbox{\reflectbox{E}}\kern-.1667em\relax}%
1955 \LaTeX}}

```

As you see, if  $\TeX$  doesn't recognize `\reflectbox` (graphics isn't loaded), the first E will not be reversed. This version of the command is intended for non- $\XeTeX$  usage. With  $\XeTeX$ , you can load the `xltxtra` package (e.g. with the `gmutils\XeTeXthree` declaration) and then the reversed E you get as the Unicode Latin Letter Reversed E.

## Expanding turning stuff all into 'other'

While typesetting a unicode file contents with `inputenc` package I got a trouble with some Unicode sequences that expanded to unexpandable CSs: they could'nt be used within `\csname... \endcsname`. My  $\TeX$ Guru advised to use `\meaning` to make all the name 'other'. So—here we are.

Don't use them in `\edefs`, they would expand not quite.

The next macro is intended to be put in `\edefs` with a macro argument. The meaning of the macro will be made all 'other' and the words '(long) macro:->' gobbled.

```

\all@other 1986 \def\all@other#1{\@xa\gm@gobmacro\meaning#1}

```

The `\gm@gobmacro` macro above is applied to gobble the `\meaning's` beginnig, long macro:-> all 'other' that is. Use of it:

```

1991 \edef\gmu@reserveda{%
\@nx 1992 \def\@nx\gm@gobmacro##1\@xa\@gobble\string\macro:->{}}
\gm@gobmacro 1993 \gmu@reserveda

```

In the next two macros' names, 'unex' stands both for not expanding the argument(s) and for disastrously partial unexpandability of the macros themselves.

```

\unex@namedef 1999 \long\def\unex@namedef#1#2{%
2000 \edef@other\gmu@reserveda{#1}%
2001 \@xa\long\@xa\def\csname\gmu@reserveda\endcsname{#2}}

\unex@nameuse 2004 \long\def\unex@nameuse#1{%
2005 \edef@other\gmu@reserveda{#1}%
2006 \csname\gmu@reserveda\endcsname}

```

## Brave New World of XeTeX

```

\ifXeTeX 2011 \newcommand\ifXeTeX[2]{%
2012   \ifdefined\XeTeXversion
2013   \unless\ifx\XeTeXversion\relax\afterfifi{#1}\else\afterfifi{%
        #2}\fi
2014   \else\afterfi{#2}\fi}

\XeTeXthree 2017 \def\XeTeXthree{%
2018   \@ifXeTeX{%
2019     \@ifpackageloaded{gmverb}{\StoreMacro\verb}{}%
2020     \RequirePackage{xltextra}% since v 0.4 (2008/07/29) this package rede-
        fines \verb and verbatim*, and quite elegantly provides an option to
        suppress the redefinitions, but unfortunately that option excludes also
        a nice definition of \xxt@visible space which I fancy.
2021     \@ifpackageloaded{gmverb}{\RestoreMacro\verb}{}%
2022     \AtBeginDocument{%
2023       \RestoreMacro\LaTeX\RestoreMacro*{LaTeX_}}% my version of the LATEX
        logo has been stored just after defining, in line 1882.
2024   }{}}
2025 }

```

The `\udigits` declaration causes the digits to be typeset uppercase. I provide it since by default I prefer the lowercase (nautical) digits.

```

2045 \AtBeginDocument{%
2046   \@ifpackageloaded{fontspec}{%
\udigits 2047   \DeclareRobustCommand*\udigits{%
2048     \addfontfeature{Numbers=Uppercase}}%
2049   }{%
2050     \emptify\udigits}}

```

## Fractions

```

\Xedekfrac 2055 \def\Xedekfracc{\@ifstar\gmu@xedekfraccstar\gmu@xedekfraccplain}

2056 (plain) The starless version turns the font feature frac on. (*) But nor Minion GM
2057 neither TEX Gyre Pagella doesn't feature the frac font feature properly so, with the
2058 starred version of the declaration we use the characters from the font where available
2059 (see the \@namedefs below) and the numr and dnom features with the fractional slash
2060 otherwise (via \gmu@dekfracc). (**) But Latin Modern Sans Serif Quotation doesn't
2061 support the numerator and denominator positions so we provide the double star ver-
2062 sion for it, which takes the char from font if it exist and typesets with lowers and kerns
2063 otherwise.

\gmu@xedekfraccstar 2069 \def\gmu@xedekfraccstar{%
\gmu@xefracccdef 2070   \def\gmu@xefracccdef##1##2{%
2071     \iffontchar\font_##2
2072     \@namedef{gmu@xefraccc##1}{\char##2_}%
2073     \else
2074     \n@melet{gmu@xefraccc##1}{relax}%
2075     \fi}%
\gmu@dekfracc 2077 \def\gmu@dekfracc##1/##2{%
2078   {\addfontfeature{VerticalPosition=Numerator}##1}%
        \gmu@numeratorkern
2079   \char"2044_\gmu@denominatorkern
2080   {\addfontfeature{VerticalPosition=Denominator}##2}}%

```

We define the fractional macros. Since Adobe Minion Pro doesn't contain  $\frac{n}{5}$  nor  $\frac{n}{6}$ , we don't provide them here.

```

2084 \gmu@xfracccdef{1/4}{BC}%
2085 \gmu@xfracccdef{1/2}{BD}%
2086 \gmu@xfracccdef{3/4}{BE}%
2087 \gmu@xfracccdef{1/3}{2153}%
2088 \gmu@xfracccdef{2/3}{2154}%
2089 \gmu@xfracccdef{1/8}{215B}%
2090 \gmu@xfracccdef{3/8}{215C}%
2091 \gmu@xfracccdef{5/8}{215D}%
2092 \gmu@xfracccdef{7/8}{215E}%
\dekfracc 2093 \def\dekfracc##1/##2{%
\gm@duppa 2094 \def\gm@duppa{##1/##2}%
2095 \@ifundefined{gmu@xfracc\all@other\gm@duppa}{%
2096 \gmu@dekfracc{##1}/{##2}}{%
2097 \csname_gmu@xfracc\all@other\gm@duppa\endcsname}}%
2098 \@ifstar{\let\gmu@dekfracc\gmu@dekfraccsimple}{}%
2099 }
\gmu@xedekfraccplain 2101 \def\gmu@xedekfraccplain{% 'else' of the main \@ifstar
\dekfracc 2102 \def\dekfracc##1/##2{%
2103 \addfontfeature{Fractions=On}%
2104 ##1/##2}}%
2105 }
\gmu@numeratorkern 2107 \def\gmu@numeratorkern{\kern-.05em\relax}
2108 \let\gmu@denominatorkern\gmu@numeratorkern

```

What have we just done? We defined two versions of the `\XeFractions` declaration. The starred version is intended to make use only of the built-in fractions such as  $\frac{1}{2}$  or  $\frac{7}{8}$ . To achieve that, a handful of macros is defined that expand to the Unicodes of built-in fractions and `\dekfracc` command is defined to use them.

The unstarred version makes use of the Fraction font feature and therefore is much simpler.

Note that in the first argument of `\@ifstar` we wrote 8 (eight) #s to get the correct definition and in the second argument 'only' 4. (The L<sup>A</sup>T<sub>E</sub>X<sub>2<sub>ε</sub></sub> Source claims that that is changed in the 'new implementation' of `\@ifstar` so maybe it's subject to change.)

A simpler version of `\dekfracc` is provided in line 2491

```

\resizegraphics
2131 \@ifXeTeX{%
\resizegraphics 2132 \def\resizegraphics#1#2#3{%
2133 \setboxo=\hbox{\XeTeXpicfile_#3}%
2134 \ifx!#1\else
2135 \dimeno=#1\relax
2136 \count2=\wdo
2137 \divide\count2_by1000\relax
2138 \counto=\dimeno\relax
2139 \divide\counto\count2
2140 \fi
2141 \ifx!#2\else
2142 \dimeno=#1\relax
2143 \count6=\hto

```

```

2144         \divide\count6_\by1000\relax
2145         \count4=\dimeno\relax
2146         \divide\count4\count6
2147         \fi
2148         \ifx!#1\counto=\count4\fi
2149         \ifx!#2\count4=\counto\fi
2150         \XeTeXpicfile_\#3_xscaled_\counto_yscaled_\count4
2151     } } } { %
\resizegraphics 2152     \def\resizegraphics#1#2#3{%
2153         \resizebox{#1}{#2}{%
2154             \includegraphics{#3}}}%

The [options] in the \XeTeXpicfile command use the following keywords:
width <dimen>
height <dimen>
scaled <scalefactor>
xscaled <scalefactor>
yscaled <scalefactor>
rotated <degrees>

\GMtextsuperscript 2165 \def\GMtextsuperscript{%
2166     \@ifXeTeX{%
\textsuperscript 2167         \def\textsuperscript##1{%
2168             \addfontfeature{VerticalPosition=Numerator}##1}}%
2169     }{\truetextsuperscript}}

\truetextsuperscript 2171 \def\truetextsuperscript{%
\textsuperscript 2172     \DeclareRobustCommand*\textsuperscript[1]{%
2173         \@textsuperscript{\selectfont##1}}%
\@textsuperscript 2174     \def\@textsuperscript##1{%
2175         {\m@th\ensuremath{\sim\mbox{\fontsize\sf@size\z@##1}}}}

```

## Varia

A very neat macro provided by doc. I copy it ~verbatim.

```

\gmu@tilde 2187 \def\gmu@tilde{%
2188     \leavevmode\lower.8ex\hbox{$_,\widetilde{\mbox{ }}\backslash,\$}}

```

Originally there was just `\` instead of `\mbox{ }` but some commands of ours do redefine `\`.

```

\* 2192 \DeclareRobustCommand*\*{\gmu@tilde}
2198 \AtBeginDocument{% to bypass redefinition of \~ as a text command with various
    encodings
\texttilde 2200 \DeclareRobustCommand*\texttilde{%
2203     \@ifnextchar/{\gmu@tilde\kern-o,1667em\relax}\gmu@tilde}}

```

We prepare the proper kerning for “~/”.

The standard `\obeyspaces` declaration just changes the space’s `\catcode` to 13 (‘active’). Usually it is fairly enough because no one ‘normal’ redefines the active space. But we are *not* normal and we do *not* do usual things and therefore we want a declaration that not only will `\activate` the space but also will (re)define it as the `\_` primitive. So define `\gmobeyspaces` that obeys this requirement.

(This definition is repeated in `gmverb`.)

```

2215 \foone{\catcode`\ \active}%

```



```
\gmobeyspaces 2216 {\def\gmobeyspaces{\let\ \catcode\ \active}}
```

While typesetting poetry, I was surprised that sth. didn't work. The reason was that original \obeylines does \let not \def, so I give the latter possibility.

```
\defobeylines 2223 \foone{\catcode\^~M\active}% the comment signs here are crucial.
2224 {\def\defobeylines{\catcode\^~M=13\def^~M{\par}}}
```

Another thing I dislike in L<sup>A</sup>T<sub>E</sub>X yet is doing special things for \...skip's, 'cause I like the Knuthian simplicity. So I sort of restore Knuthian meanings:

```
\dekssmallskip 2233 \def\dekssmallskip{\vskip\smallskipamount}
\undeekssmallskip 2234 \def\undeekssmallskip{\vskip-\smallskipamount}
\dekmedskip 2235 \def\dekmedskip{\vskip\medskipamount}
\dekbigskip 2236 \def\dekbigskip{\vskip\bigskipamount}
\hfillneg 2239 \def\hfillneg{\hskip\opt\plus\ifill\relax}
```

In some \if(cat?) test I needed to look only at the first token of a tokens' string (first letter of a word usually) and to drop the rest of it. So I define a macro that expands to the first token (or {\text}) of its argument.

```
\@firstofmany 2247 \long\def\@firstofmany#1#2\@nil{#1}
```

A mark for the **TODO!**s:

```
\TODO 2251 \newcommand*{\TODO}[1] []{ {%
2252 \sffamily\bfseries\huge\TODO!\if\relax#1\relax\else\space%
\fi#1}}
```

I like twocolumn tables of contents. First I tried to provide them by writing \begin{%multicols}{2} and \end{multicols} outto the .toc file but it worked wrong in some cases. So I redefine the internal L<sup>A</sup>T<sub>E</sub>X macro instead.

```
\twocoltoc 2287 \newcommand*\twocoltoc{%
2288 \RequirePackage{multicol}%
\starttoc 2289 \def\@starttoc##1{%
2290 \begin{multicols}{2}\makeatletter\@input{\jobname.##1}%
2291 \if@filesw\@xa\newwrite\csname_tfc##1\endcsname
2292 \immediate\openout\csname_tfc##1\endcsname\jobname.
##1\relax
2293 \fi
2294 \@nobreakfalse\end{multicols}}
2296 \@onlypreamble\twocoltoc
```

The macro given below is taken from the multicol package (where its name is \enough@room). I put it in this package since I needed it in two totally different works.

```
\enoughpage 2302 \newcommand\enoughpage[1]{%
2303 \par
2304 \dimeno=\pagegoal
2305 \advance\dimeno by-\pagetotal
2306 \ifdim\dimeno<#1\relax\newpage\fi}
```

Two shorthands for debugging:

```
\tOnLine 2310 \newcommand*\tOnLine{\typeout{\on@line}}
\OnAtLine 2312 \let\OnAtLine\on@line
```

An equality sign properly spaced:

```
\equals 2316 \newcommand*\equals{${}={}${}}
```

And for the L<sup>A</sup>T<sub>E</sub>X's pseudo-code statements:

```
\eequals 2318 \newcommand*\eequals{${}=={}}$}
```

While typesetting a UTF-8 ls-R result I found a difficulty that follows: UTF-8 encoding is handled by the inputenc package. It's O.K. so far. The UTF-8 sequences are managed using active chars. That's O.K. so far. While writing such sequences to a file, the active chars expand. You feel the blues? When the result of expansion is read again, it sometimes is again an active char, but now it doesn't star a correct UTF-8 sequence.

Because of that I wanted to 'freeze' the active chars so that they would be \written to a file unexpanded. A very brutal operation is done: we look at all 256 chars' catcodes and if we find an active one, we \let it \relax. As the macro does lots and lots of assignments, it shouldn't be used in \edefs.

```
\freeze@actives 2338 \def\freeze@actives{%
2339   \count\z@\z@
2341   \@whilenum\count\z@<\@cclvi\do{%
2342     \ifnum\catcode\count\z@=\active
~ 2343       \uccode`~=\count\z@
2344       \uppercase{\let~\relax}%
2345     \fi
2346     \advance\count\z@\@ne}}
```

A macro that typesets all 256 chars of given font. It makes use of \@whilenum.

```
\ShowFont 2352 \newcommand*\ShowFont[1][6]{%
2353   \begin{multicols}{#1}[The\currentfont\the\fontencoding%
    \ encoding):]
2354   \parindent\z@
2355   \count\z@\m@ne
2356   \@whilenum\count\z@<\@cclv\do{
2357     \advance\count\z@\@ne
2358     \ \the\count\z@:\~\char\count\z@\par}
2359   \end{multicols}}
```

A couple of macros for typesetting liturgic texts such as psalmody of Liturgia Horarum. I wrap them into a declaration since they'll be needed not every time.

```
\liturgiques 2367 \newcommand*\liturgiques[1][red]{% Requires the color package.
2368   \gmu@RPif{color}{color}%
\czerwo 2369   \newcommand*\czerwo{\small\color{#1}}% environment
\czer 2370   \newcommand{\czer}[1]{\leavevmode{\czerwo##1}}% we leave vmode be-
    cause if we don't, then verse's \everypar would be executed in a group
    and thus its effect lost.
~ 2373   \def\*{\czer{*$*$}}
+ 2374   \def\+{\czer{$\dag$}}
\nieczer 2375   \newcommand*\nieczer[1]{\textcolor{black}{##1}}}
```

After the next definition you can write \gmu@RP[options]{package}{csname} to get the package #2 loaded with options #1 if the csname #3 is undefined.

```
\gmu@RPif 2380 \newcommand*\gmu@RPif[3][ ]{%
2381   \ifx\relax#1\relax
\gmu@resa 2382   \else\def\gmu@resa{[#1]}%
2383   \fi
2384   \@xa\RequirePackage\gmu@resa{#2}}
```

Since inside document we cannot load a package, we'll redefine \gmu@RPif to issue a request before the error issued by undefined CS.

```

2390 \AtBeginDocument{%
\gmu@RPif 2391 \renewcommand*\gmu@RPif[3][{}]{%
2392 \ifundefined{#3}{%
2393 \ifpackageloaded{#2}{%
2394 \typeout{^^J!_Package_`#2'_not_loaded!!!_(%
\on@line)^^J}}{}}}}

```

It's very strange to me but it seems that `c` is not defined in the basic math packages. It is missing at least in the *Symbols* book.

```

\continuum 2400 \providecommand*\continuum{\gmu@RPif{eufrak}{mathfrak}{\mathfrak{c}}
c}}

```

And this macro I saw in the `ltugproc` document class and I liked it.

```

\iteracro 2404 \def\iteracro{%
\acro 2405 \DeclareRobustCommand*\acro[1]{\gmu@acrospaces##1_
\gmu@acrospaces}%
2406 }
2408 \iteracro
\gmu@acrospaces 2410 \def\gmu@acrospaces#1_#2\gmu@acrospaces{%
2411 \gmu@acroinner#1\gmu@acroinner
2412 \ifx\relax#2\relax\else
2413 \space
2414 \afterfi{\gmu@acrospaces#2\gmu@acrospaces}% when #2 is nonempty, it
is ended with a space. Adding one more space in this line resulted in an
infinite loop.
2418 \fi}
\gmu@acroinner 2421 \def\gmu@acroinner#1{%
2422 \ifx\gmu@acroinner#1\relax\else
2423 \ifcat_a\@nx#1\relax%
2424 \ifnum`#1=\uccode`#1%
2425 {\acrocore{#1}}%
2426 \else{#1}% tu bylo \smallerr
2427 \fi
2428 \else#1%
2429 \fi
2430 \afterfi\gmu@acroinner
2431 \fi}

```

We extract the very thing done to the letters to a macro because we need to redefine it in fonts that don't have small caps.

```

\acrocore 2435 \def\acrocore{\scshape\lowercase}

```

Since the fonts I am currently using do not support required font feature, I skip the following definition.

```

\IMO 2440 \newcommand*\IMO{\acro{IMO}}
\AKA 2441 \newcommand*\AKA{\acro{AKA}}
\usc 2443 \DeclareRobustCommand*\usc[1]{\addfontfeature{
Letters=UppercaseSmallCaps}#1}}
\uscacro 2445 \def\uscacro{\let\acro\usc}
\qxenc 2447 \newcommand*\qxenc{\fontencoding{QX}\selectfont}

```

The `\copyright` command is unavailable in T1 and U (unknown) encodings so provide

```

\qxcopyright 2450 \newcommand*\qxcopyright{\qxcenc\copyright}}
\qxcopyrights 2451 \newcommand*\qxcopyrights{%
2452 \let\gmu@copyright\copyright
2453 \def\copyright{\qxcenc\gmu@copyright}}

```

```

\fixcopyright 2455 \newcommand*\fixcopyright{%
2456 \@ifXeTeX{\def\copyright{\char"00A9}}{\qxcopyrights}}

```

Probably the only use of it is loading gmdocc.cls ‘as second class’. This command takes first argument optional, options of the class, and second mandatory, the class name. I use it in an article about gmdoc.

```

\secondclass 2463 \def\secondclass{%
\ifSecondClass 2464 \newif\ifSecondClass
2465 \SecondClasstrue
2466 \@fileswithoptions\@clsextension}% [outeroff,gmeometric]{gmdocc}
it’s loading gmdocc.cls with all the bells and whistles except the error mes-
sage.

```

Cf. *The T<sub>E</sub>Xbook* exc. 11.6.

A line from L<sup>A</sup>T<sub>E</sub>X:

```
% \check@mathfonts\fontsize\sf@size\z@\math@fontsfalse\selectfont
```

didn’t work as I would wish: in a \footnotesize’s scope it still was \scriptsize, so too large.

```

\gmu@dekfracssimple 2484 \def\gmu@dekfracssimple#1/#2{\leavevmode\kern.1em
2485 \raise.5ex\hbox{\udigits\smaller[3]#1}\gmu@numeratorkern
2486 \dekfracslash\gmu@denominatorkern
2488 {\udigits\smaller[3]#2}}%

```

```

\dekfracssimple 2491 \def\dekfracssimple{%
2492 \let\dekfrac\gmu@dekfracssimple
2493 }

```

```
\dekfracslash 2494 \@ifXeTeX{\def\dekfracslash{\char"2044}}{%
```

```
\dekfracslash 2495 \def\dekfracslash{/}}\char"2044
```

```
2497 \dekfracssimple
```

A macro that acts like \, (thin and unbreakable space) except it allows hyphenation afterwards:

```
\ikern 2505 \newcommand*\ikern{\,\penalty10000\hskiposp\relax}
```

And a macro to forbid hyphenation of the next word:

```
\nohy 2509 \newcommand*\nohy{\leavevmode\kernosp\relax}
```

```
\yeshy 2510 \newcommand*\yeshy{\leavevmode\penalty10000\hskiposp\relax}
```

In both of the above definitions ‘osp’ not \z@ to allow their writing to and reading from files where @ is ‘other’.

```
\@ifempty
```

```

\@ifempty 2516 \long\def\@ifempty#1#2#3{%
\gmu@reserveda 2517 \def\gmu@reserveda{#1}%
2518 \ifx\gmu@reserveda\empty\afterfi{#2}%
2519 \else\afterfi{#3}\fi
2520 }

```

`\include not only .tex's`

`\include` modified by me below lets you to include files of any extension provided that extension in the argument.

If you want to `\include` a non-`.tex` file and deal with it with `\includeonly`, give the latter command full file name, with the extension that is.

```
\gmu@gettext 2532 \def\gmu@gettext#1.#2\@nil{%
\gmu@filename 2533 \def\gmu@filename{#1}%
\gmu@fileext 2534 \def\gmu@fileext{#2}}

2536 \def\include#1{\relax
2537 \ifnum\@auxout=\@partaux
2538 \latexerror{\string\include\space cannot be nested}\@eha
2539 \else\@include#1\fi}

\include 2541 \def\@include#1_{%
2542 \gmu@gettext#1.\@nil
\gmu@fileext 2543 \ifx\gmu@fileext\empty\def\gmu@fileext{tex}\fi
2544 \clearpage
2545 \if@files
2546 \immediate\write\@mainaux{\string\@input{\gmu@filename.aux}}%
2547 \fi
2548 \@tempwattrue
2549 \if@partsw
2550 \@tempswafalse
2551 \edef\reserved@b{#1}%
2552 \@for\reserved@a:=\@partlist\do{%
2553 \ifx\reserved@a\reserved@b\@tempwattrue\fi}%
2554 \fi
2555 \if@tempswa
2556 \let\@auxout\@partaux
2557 \if@files
2558 \immediate\openout\@partaux\gmu@filename.aux
2559 \immediate\write\@partaux{\relax}%
2560 \fi
2561 \@input{\gmu@filename.\gmu@fileext}%
2562 \inclasthook
2563 \clearpage
2564 \@writeckpt{\gmu@filename}%
2565 \if@files
2566 \immediate\closeout\@partaux
2567 \fi
2568 \else

If the file is not included, reset \@include \deadcycles, so that a long list of non-
included files does not generate an 'Output loop' error.

2572 \deadcycles\z@
2573 \@nameuse{cp@\gmu@filename}%
2574 \fi
2575 \let\@auxout\@mainaux}

\whenonly 2578 \newcommand\whenonly[3]{%
\gmu@whonly 2579 \def\gmu@whonly{#1,}%
2580 \ifx\gmu@whonly\@partlist\afterfi{#2}\else\afterfi{#3}\fi
```

I assume one usually includes chapters or so so the last page style should be closing.

```

\inclasthook 2584 \def\inclasthook{\thispagestyle{closing}}

Faked small caps

\gmu@scapLetters 2590 \def\gmu@scapLetters#1{%
2591   \ifx#1\relax\relax\else% two \relaxes to cover the case of empty #1.
2592   \ifcat_\a#1\relax
2593   \ifnum\the\lccode`#1=`#1\relax
2594   {\fakescapsscore\MakeUppercase{#1}}}% not Plain \uppercase because
        that works bad with inputenc.
2596   \else#1%
2597   \fi
2598   \else#1%
2599   \fi%
2600   \@xa\gmu@scapLetters
2601   \fi}%
\gmu@scapSpaces 2603 \def\gmu@scapSpaces#1_\#2\@@nil{%
2604   \ifx#1\relax\relax
2605   \else\gmu@scapLetters#1\relax
2606   \fi
2607   \ifx#2\relax\relax
2608   \else\afterfi{\ \gmu@scapSpaces#2\@@nil}%
2609   \fi}
\gmu@scapss 2611 \def\gmu@scapss#1\@@nil{{\def~{{\nobreakspace}}}%
\nobreakspace 2612   \gmu@scapSpaces#1_\@@nil}}}% \def\\{{\newline}}\relax adding re-
        definition of \\ caused stack overflow Note it disallows hyphenation ex-
        cept at \-.
\fakecaps 2616 \DeclareRobustCommand\fakecaps[1]{{%
2617   \gmu@scapss#1\@@nil}}
2619 \let\fakecapscore\gmu@scalematchX
        Experimente z akcentami patrz no3.tex.
\tinycapae 2622 \def\tinycapae{{\tiny\AE}}}% to use in \fakecaps[\tiny]{...}
2624 \RequirePackage{calc}
        wg \zf@calc@scale pakietu fontspec.
2628 \@ifXeTeX{%
\gmu@scalar 2629   \def\gmu@scalar{1.0}%
\zf@scale 2630   \def\zf@scale{}}%
\gmu@scalematchX 2631   \def\gmu@scalematchX{%
2632     \begingroup
\gmu@scalar 2633     \ifx\zf@scale\empty\def\gmu@scalar{1.0}%
2634     \else\let\gmu@scalar\zf@scale\fi
2635     \setlength\@tempdima{\fontdimen5\font}% 5—ex height
2636     \setlength\@tempdimb{\fontdimen8\font}% 8—XTeX synthesized up-
        percase height.
2638     \divide\@tempdimb_\by1000\relax
2639     \divide\@tempdima_\by\@tempdimb
2640     \setlength{\@tempdima}{\@tempdima*\real{\gmu@scalar}}}%
2641     \@ifundefined{fakesc@extrascap}{\setlength{\@tempdima}{\@tempdima*\real{\gmu@scalar}}}%
2642     {\setlength{\@tempdima}{\@tempdima*\real{\gmu@scalar}}}%
2643     \@tempcnta=\@tempdima

```

```

2644 \divide\@tempcnta by 1000\relax
2645 \@tempcntb=-1000\relax
2646 \multiply\@tempcntb by \@tempcnta
2647 \advance\@tempcntb by \@tempdima
2648 \xdef\gmu@scscale{\the\@tempcnta.%
2649 \ifnum\@tempcntb<1000\fi
2650 \ifnum\@tempcntb<1000\fi
2651 \the\@tempcntb}%
2652 \endgroup
2653 \addfontfeature{Scale=\gmu@scscale}%
2654 }}{\let\gmu@scalematchX\smallerr}
\fakecextrasc 2657 \def\fakecextrasc#1{\def\fakec@extrasc{#1}}
\fakec@extrasc

```

### See above/see below

To generate a phrase as in the header depending of whether the respective label is before or after.

```

\wyzejnizej 2663 \newcommand*\wyzejnizej[1]{%
2664 \edef\gmu@tempa{\ifundefined{r@#1}{\arabic{page}}}%
2665 \@xa\@xa\@xa\@secondoftwo\csname r@#1\endcsname}%
2666 \ifnum\gmu@tempa<\arabic{page}\relax wy\zej\fi
2667 \ifnum\gmu@tempa>\arabic{page}\relax ni\zej\fi
2668 \ifnum\gmu@tempa=\arabic{page}\relax \@xa\ignorespaces\fi
2669 }

```

### luzniej and napapierki—environments used in page breaking for money

The name of first of them comes from Polish typesetters’ phrase “rozbijać [skład] na papierki”—‘to broaden [leading] with paper scratches’.

```

\napapierkistretch 2679 \def\napapierkistretch{0,3pt}% It’s quite much for 11/13pt typesetting
\napapierkicore 2681 \def\napapierkicore{\advance\baselineskip%
2682 by\optplus\napapierkistretch\relax}
napapierki 2684 \newenvironment*{napapierki}{%
2685 \par\global\napapierkicore}%
2686 \par\dimen\z@=\baselineskip
2687 \global\baselineskip=\dimen\z@}% so that you can use \endnapapierki in
interlacing environments

```

```

\gmu@luzniej 2691 \newcount\gmu@luzniej
\luzniejcore 2693 \newcommand*\luzniejcore[1][1]{%
2694 \advance\gmu@luzniej\@ne% We use this count to check whether we open the
environment or just set \looseness inside it again.
2696 \ifnum\gmu@luzniej=\@ne\multiply\tolerance by 2\fi
2697 \looseness=#1\relax}

```

After `\begin{luzniej}` we may put the optional argument of `\luzniejcore`

```
luzniej 2701 \newenvironment*{luzniej}{\par\luzniejcore}{\par}
```

The starred version does that `\everypar`, which has its advantages and disadvantages.

```
luzniej* 2706 \newenvironment*{luzniej*}[1][1]{%
2707 \multiply\tolerance by 2\relax

```

```
2708 \everypar{\looseness=#1\relax}}{\par}
```

\nawj 2710 \newcommand\*\nawj{\kerno,1em\relax}% to put between parentheses and letters with lower ... such as *j* or *y* in certain fonts.

The original \pauza of polski has the skips rigid (one is even a kern). It begins with \ifhmode to be usable also at the beginning of a line as the mark of a dialogue.

```
2717 \ifdefined\XeTeXversion
2718 \AtBeginDocument{% to be independent of moment of loading of polski.
\pauzacore 2719 \DeclareRobustCommand*\pauza{%
2720 \ifhmode
2721 \unskip\penalty10000
2722 \afterfi{%
2723 \@ifnextspace{\hskipo.2em\pluso.1em\relax
2724 \pauzacore\hskip.2em\pluso.1em\relax\ignorespaces}%
2725 {\pauzacore\penalty\hyphenpenalty\hskip\z@}}}%
2726 \else
```

According to *Instrukcja technologiczna. Skład ręczny i maszynowy* the dialogue dash should be followed by a rigid hskip of ½ em.

```
2730 \leavevmode\pauzacore\penalty10000\hskipo,5em\ignorespaces
2731 \fi}%
```

The next command's name consists of letters and therefore it eats any spaces following it, so \@ifnextspace would always be false.

```
\pauza 2734 \DeclareRobustCommand*\pauza{%
2735 \ifhmode
2736 \unskip\penalty10000
2737 \hskipo.2em\pluso.1em\relax
2738 \pauzacore\hskip.2em\pluso.1em\relax\ignorespaces%
2739 \else
```

According to *Instrukcja technologiczna. Skład ręczny i maszynowy* the dialogue dash should be followed by a rigid hskip of ½ em.

```
2743 \leavevmode\pauzacore\penalty10000\hskipo,5em\ignorespaces
2744 \fi}%
```

And a version with no space at the left, to begin a \noindent paragraph or a dialogue in quotation marks:

```
\lpauza 2747 \DeclareRobustCommand*\lpauza{%
2748 \pauzacore\hskip.2em\pluso.1em\ignorespaces}%
```

We define \ppauza as an en dash surrounded with thin stretchable spaces and sticking to the upper line or bare but discretionary depending on the next token being space<sub>10</sub>. Of course you'll never get such a space after a literal CS so an explicit \ppauza will always result with a bare discretionary en dash, but if we \let-\ppauza...

```
\- 2756 \DeclareRobustCommand*\ppauza{%
2757 \ifvmode\PackageError{gmutils}{%
2758 command\backslashppauza(en dash) not intended for vmode.}%
2759 Use\backslashppauza(en dash) only in number and numeral
ranges.}%
2760 \else
2761 \afterfi{%
2762 \@ifnextspace{\unskip\penalty10000\hskipo.2em\pluso.1em%
\relax
```



```

2763         -\hskip.2em\pluso.1em\ignorespaces}{\unskip%
          \discretionary{-}{-}{-}}}%
2764     \fi}%
\ppauza 2766     \DeclareRobustCommand*\ppauza{%
2767     \ifvmode\PackageError{gmutils}{%
2768     command\backslashppauza(en_dash)not_intended_for_vmode.}{%
2769     Use\backslashppauza(en_dash)only_in_number_and_numeral_
          ranges.}%
2770     \else
2771     \unskip\discretionary{-}{-}{-}%
2772     \fi}%
\emdash 2774     \def\emdash{\char`-}
2775     }% of at begin document
\longpauza 2777 \def\longpauza{\def\pauzacore{-}}
\pauzacore 2778 \longpauza
\shortpauza 2779 \def\shortpauza{%
\pauzacore 2780     \def\pauzacore{-\kern,23em\relax\llap{-}}}%
2781 \fi% of if XeTeX.

```

If you have all the three dashes on your keyboard (as I do), you may want to use them for short instead of \pauza, \ppauza and \dywiz. The shortest dash is defined to be smart in math mode and result with  $-$ .

```

2787 \ifdefined\XeTeXversion
2788 \foone{\catcode`\-active\catcode`\-active\catcode`\-active}{%
\adashes 2789     \def\adashes{\AtBeginDocument\adashes}% because \pauza is defined at
          begin document.
\adashes 2791     \AtBeginDocument{\def\adashes{%
2792     \catcode`\-active\let-}%
2793     \catcode`\-active\let-}%
2795     }}}
2796 \else
2797 \relaxen\adashes
2798 \fi

```

The hyphen shouldn't be active IMO because it's used in TeX control such as \hskip-2pt. Therefore we provide the \ahyphen declaration reluctantly, because sometimes we need it and always use it with caution. Note that my active hyphen in vertical and math modes expands to  $-_{12}$ .

```

\gmu@dywiz 2807 \def\gmu@dywiz{\ifmmode-\else
2808     \ifvmode-\else\afterfifi\dywiz\fi\fi}%
2810 \foone{\catcode`\-active}{%
\ahyphen 2811     \def\ahyphen{\let-\gmu@dywiz\catcode`\-active}}
          To get current time. Works in  $\varepsilon$ -TeXs, including XeTeX.
\czas 2815 \newcommand*\czas[1][.]{%
2816     \the\numexpr(\time-30)/60\relax#1%
2817     \@tempcnta=\numexpr\time-(\time-30)/60*60\relax
2818     \ifnum\@tempcnta<10\fi\the\@tempcnta}
          To push the stuff up to the header and have the after heading skip after the stuff
\przeniesvskip 2823 \long\def\przeniesvskip#1{%
2824     \edef\gmu@LastSkip{\the\lastskip}%
2825     \vskip-\gmu@LastSkip\relax

```

	2826	\vspace*{osp}%
	2827	#1\vskip\gmu@LastSkip\relax}
\textbullet	2829	\@ifXeTeX{\chardef\textbullet="2022□}{\def\textbullet{\$\bullet\$}}
tytulowa	2831	\newenvironment*{tytulowa}{\newpage}{\par\thispagestyle{empty}% \newpage}
		Nazwisko na stronę redakcyjną
\nazwired	2834	\def\nazwired{\quad\textsc}

## Settings for mathematics in main font

I used this terrible macros while typesetting E. Szarzyński's *Letters* in 2008.

```

\gmath 2839 \def\gmath{%
2840   \def\do##1{\edef##1{\{\@nx\mathit{\@xa\@gobble\string##1}\}}}%
2841   \do\A_\do\A_\do\B_\do\B_\do\C_\do\C_\do\D_\do\E_\do\E_\do\F
2842   \do\F_\do\G_\do\G_\do\I_\do\I_\do\J_\do\J_\do\K_\do\K_\do\L_\do\L_\do\m
2843   \do\M_\do\N_\do\N_\do\P_\do\P_\do\Q_\do\Q_\do\R_\do\R
2844   \let\sectionsign\S_\do\S_\do\S_\do\T_\do\T_\do\U_\do\U_\do\V
2845   \do\W_\do\W_\do\X_\do\X_\do\Y_\do\Y_\do\Z_\do\Z
2847   \def\do##1{\edef##1{\{\@nx\mathrm{\@xa\@gobble\string##1}\}}}%
2848   \do\o\do\1\do\2\do\3\do\4\do\5\do\6\do\7\do\8\do\9%
2850   \relaxen\do
2851   \newcommand*\do[4][\mathit]{\def##2{##3{##1{\char"##4}}}}%
2852   \do\alpha{}{03B1}%
2853   \do[\mathrm]\Delta{}{0394}%
2854   \do\varepsilon{}{03B5}%
2855   \do\vartheta{}{03D1}%
2856   \do\nu{}{03BD}%
2857   \do\pi{}{03C0}%
2858   \do\phi{}{03D5}%
2859   \do[\mathrm]\Phi{}{0424}%
2860   \do\sigma{}{03C3}%
2861   \do\varsigma{}{03DA}%
2862   \do\psi{}{03C8}%
2863   \do\omega{}{03C9}%
2864   \do\infty{}{221E}%
2865   \do[\mathrm]\neg{\mathbin}{00AC}%
2866   \do[\mathrm]\neq{\mathrel}{2260}%
2867   \do\partial{}{2202}%
2868   \do[\mathrm]\pm{}{00B1}%
2869   \do[\mathrm]\pm{\mathbin}{00B1}%
2870   \do[\mathrm]\sim{\mathrel}{007E}%
2872   \def\do##1##2##3{\def##1{%
\mathop{\mathchoice{\hbox{%
2873     \rm
2874     \edef\gma@tempa{\the\fontdimen8\font}%
2875     \larger[3]%
2876     \lower\dimexpr(\fontdimen8\font-\gma@tempa)/2_\do
2877     \hbox{##2}}}{\hbox{%
2878

```

```

2879         \rm
2880         \edef\gma@tempa{\the\fontdimen8\font}%
2881         \larger[2]%
2882         \lower\dimexpr(\fontdimen8\font-\gma@tempa)/2\relax
2883         \hbox{##2}}}%
2884         {\mathrm{##2}}{\mathrm{##2}}##3}}%
2885 \do\sum{\char"2211}{}%
2886 \do\forall{\gma@quantifierhook\rotatebox[origin=c]{180}{A}%
2887   \setboxo=\hbox{A}\setbox2=\hbox{\scriptsize x}%
2888   \kern\dimexpr\ht2/3*2-\wdo/2\relax}{\nolimits}%
2889 \do\exists{\rotatebox[origin=c]{180}{\gma@quantifierhook E}}%
2890   \nolimits%
2891 \def\do##1##2##3{\def##1{##3{%
\mathchoice 2892   \mathchoice{\hbox{\rm##2}}{\hbox{\rm##2}}}%
2893   {\hbox{\rm\scriptsize##2}}{\hbox{\rm\tiny##2}}}}}%
2894   \do\vee{\rotatebox[origin=c]{90}{<}}\mathbin
2895   \do\wedge{\rotatebox[origin=c]{-90}{<}}\mathbin
2896   \do\leftarrow{\char"2190}\mathrel
2897   \do\rightarrow{\char"2192}\mathrel
2898   \do\leftrightharpoonup{\char"2190\kern-0,1em\char"2192}\mathrel
2899 \def\do##1##2##3{%
2900   \catcode`##1=12\relax
2901   \scantokens{\mathcode`##1="8000\relax
2902     \foone{\catcode`##1=\active}{\def##1}{##3{%
\mathchoice 2903       \mathchoice{\hbox{\rm##2}}{\hbox{\rm##2}}}%
2904       {\hbox{\rm\scriptsize##2}}{\hbox{\rm\tiny##2}}}}}%
2905       \ignorespaces}}% to eat the lineend (scantokens acts as \read including
2906       line end.
2907 \do..\mathpunct\do,,\mathpunct\do.....\mathpunct
2908 \do(\mathopen
2909 \do\@ifundefined{resetMathstrut@}{\% an error occurred 'bad mathchar etc.'
2910   because amsmath.sty doesn't take account of a possibility of ( ) being math-
2911   active.
\resetMathstrut@ 2912 \def\resetMathstrut@{%
2913   \setbox\z@\hbox{%
2914     %% \mathchardef\@tempa\mathcode`\(\relax%% \def\@tempb##1"##2##3{%
2915     \the\textfont"##3\char"}%% \expandafter\@tempb\meaning\@tempa \relax
2916     {}%
2917     \ht\Mathstrutbox@\ht\z@\dp\Mathstrutbox@\dp\z@
2918   }}%
2919 \do)\mathclose
2920 \do[[\mathopen\do]]\mathclose
2921 \do-{\char"2212}\mathbin\do++\mathbin\do==\mathrel\do\times%
2922   \mathbin
2923 \do::\mathbin\do\cdots\mathbin\do/\mathbin\do<<\mathrel
2924 \do>>\mathrel
2925 \def\do##1##2##3{\def##1####1{##2{\hbox{%
2926   \rm
2927   \setboxo=\hbox{####1}%
2928   \edef\gma@tempa{\the\hto}%
2929   \edef\gma@tempb{\the\dp}%
2930   ##3%

```

```

2934         \setboxo=\hbox{####1}%
2935         \lower\dimexpr(\hto_+\dpo)/2-\dpo_-(\gma@tempa+%
           \gma@tempb)/2-\gma@tempb)__%
2936         \boxo}}}}%
2937     \do\bigl\mathopen\larger
2938     \do\bigr\mathclose\larger
2939     \do\Bigl\mathopen\largerr
2940     \do\Bigr\mathclose\largerr
2941     \do\biggl\mathopen{\larger[3]}%
2942     \do\biggr\mathclose{\larger[3]}%
2943     \do\Biggl\mathopen{\larger[4]}%
2944     \do\Bigr\mathclose{\larger[4]}%
2945     \def\do##1##2{\def##1{\ifmmode##2{\mathchoice
2946         {\hbox{\rm\char`##1}}{\hbox{\rm\char`##1}}%
2947         {\hbox{\rm\scriptsize\char`##1}}{\hbox{\rm\tiny%
           \char`##1}}}%
2948         \else\char`##1\fi}}%
2949     \StoreMacros{\{\}}%
2950     \do\{\mathopen
2951     \do\}\mathclose
2952     \def\={\mathbin{=}}%
2953 \neqb 2954     \def\neqb{\mathbin{\neq}}%
2955     \def\do##1{\edef\gma@tempa{%
2956         \def\@xa\@nx\cename_\@xa\gobble\string##1r\endcsname{%
2957             \@nx\mathrel{\@nx##1}}}%
2958         \gma@tempa}%
2959     \do\vee_\do\wedge_\do\neg
2960 \fakern 2961     \def\fakern{\mkern-3mu}%
2962     \thickmuskip=8mu_plus_4mu\relax
2963     \gma@gmathhook
2964 }% of def gmath
2965
2966 \quantifierhook 2967     \emptify\gma@quantifierhook
2968 \gma@quantifierhook 2969     \def\quantifierhook#1{%
2970         \def\gma@quantifierhook{#1}}
2971
2972 \gmathhook 2973     \emptify\gma@gmathhook
2974     \def\gmathhook#1{\addtomacro\gma@gmathhook{#1}}
2975
2976 \gma@dollar 2977     \def\gma@dollar$#1${\gmath$#1$}%
2978 \gma@bare 2979     \def\gma@bare#1{\gma@dollar$#1$}%
2980 \gma@checkbracket 2981     \def\gma@checkbracket{\@ifnextchar\[%
2982         \gma@bracket\gma@bare}
2983 \gma@bracket 2984     \def\gma@bracket[#1\]{\gmath[#1\]}\@ifnextchar\par{\{%
2985         \noindent}}
2986 \gma 2987     \def\gma{\@ifnextchar$%
2988         \gma@dollar\gma@checkbracket}
2989
2990 \garamath 2991     \def\garamath{%
2992         \quantifierhook{\addfontfeature{OpticalSize=800}}%
2993         \def\gma@arrowdash{\%
2994             \setboxo=\hbox{\char"2192}\copyo\kern-o,6\wdo
2995             \bgcolor\rule[-\dpo]{o,6\wdo}{\dimexpr\hto+\dpo}\kern-o,6%
2996             \wdo}}%
2997 \gma@gmathhook 2998     \def\gma@gmathhook{%

```

```

2995 \def\do####1####2####3{\def####1{####3{%
\mathchoice 2996 \mathchoice{\hbox{\rm####2}}{\hbox{\rm####2}}%
2997 {\hbox{\rm\scriptsize####2}}{\hbox{\rm\tiny####2}}}}}%
2998 \do\mapsto{\rule[0,4ex]{0,1ex}{0,4ex}\kern-0,05em%
2999 \gma@arrowdash\kern-0,05em\char"2192}\mathrel
3000 \do\cup{\scshape\char"2044}\mathbin
3001 \do\varnothing{\setbox0=\hbox{\gma@quantifierhook%
\addfontfeature{Scale=1.272727}0}%
3002 \setbox2=\hbox{\char"2044}}%
3003 \copy0\kern-0,5\wdo\kern-0,5\wd2\lower0,125\wdo\copy2
3004 \kern0,5\wdo\kern-0,5\wd2}{}%
3005 \do\leftarrow{\char"2190\kern-0,05em\gma@arrowdash}\mathrel
3006 \do\rightarrow{\gma@arrowdash\kern-0,05em\char"2192}\mathrel
3007 \do\in{\gma@quantifierhook\char"0454}\mathbin
3008 }}

```

## Typesetting dates in my memoirs

A date in the YYYY-MM-DD format we'll transform into DD mmmm YYYY format or we'll just typeset next two tokens/{...} if the arguments' string begins with --. The latter option is provided to preserve compatibility with already used macros and to avoid a starred version of \thedata and the same time to be able to turn \datef off in some cases (for SevSevo4.tex).

```

\polskadata 3020 \newcommand*\polskadata{%
\datef 3021 \def\datef##1-##2-##3##4{%
3022 \if\relax##2\relax##3##4%
3023 \else
3024 \ifnum##3##4=0\relax
3025 \else
3026 \ifnum##3=0\relax
3027 \else##3%
3028 \fi##4%
3029 \fi
3030 \ifcase##2\relax\or\ stycznia\or\ lutego%
3031 \or\ marca\or\ kwietnia\or\ maja\or\ czerwca\or\ lipca\or%
\ sierpnia%
3032 \or\ wrzesnia\or\ pazdziernika\or\ listopada\or\ grudnia%
\else
3033 {}%
3034 \fi
3035 \if\relax##1\relax\else\ \fi##1%
3036 \fi}%
\datefsl 3039 \def\datefsl##1/##2/##3##4{%
3040 \if\relax##2\relax##3##4%
3041 \else
3042 \ifnum##3##4=0\relax
3043 \else
3044 \ifnum##3=0\relax
3045 \else##3%
3046 \fi##4%
3047 \fi
3048 \ifcase##2\relax\or\ stycznia\or\ lutego%

```

```

3049         \or\ marca\or\ kwietnia\or\ maja\or\ czerwca\or\ lipca\or%
           \ sierpnia%
3050         \or\ września\or\ października\or\ listopada\or\ grudnia%
           \else
3051     {}%
3052     \fi
3053     \if\relax##1\relax\else\ \fi_##1%
3054 \fi}%
3055 }% of \polskadata
3057 \polskadata
    For documentation in English:
\englishdate 3060 \newcommand*\englishdate{%
\datef      3061 \def\datef##1-##2-##3##4{%
3062     \if\relax##2\relax##3##4%
3063     \else
3064         \ifcase##2\relax\or_January\or_February%
3065         \or_March\or_April\or_May\or_June\or_July\or_August%
3066         \or_September\or_October\or_November\or_December\else
3067         {}%
3068     \fi
3069     \ifnum##3##4=0\relax
3070     \else
3071         \ %
3072         \ifnum##3=0\relax
3073         \else##3%
3074         \fi##4%
3075         \ifcase##3##4\relax\or_st\or_nd\or_rd\else_th\fi
3076     \fi
3077     \if\relax##1\relax\else,\ \fi_##1%
3078 \fi
3079 }%
\datefsl      3080 \def\datefsl##1/##2/##3##4{%
3081     \if\relax##2\relax##3##4%
3082     \else
3083         \ifcase##2\relax\or_January\or_February%
3084         \or_March\or_April\or_May\or_June\or_July\or_August%
3085         \or_September\or_October\or_November\or_December\else
3086         {}%
3087     \fi
3088     \ifnum##3##4=0\relax
3089     \else
3090         \ %
3091         \ifnum##3=0\relax
3092         \else##3%
3093         \fi##4%
3094         \ifcase##3##4\relax\or_st\or_nd\or_rd\else_th\fi
3095     \fi
3096     \if\relax##1\relax\else,\ \fi_##1%
3097 \fi
3098 }%
3099 }
\ifgmu@dash 3101 \newif\ifgmu@dash

```

```

\gmu@ifnodash 3103 \def\gmu@ifnodash#1-#2\@@nil{%
3104   \def\@tempa{#2}%
3105   \ifx\@tempa\@empty}

\gmu@testdash 3107 \def\gmu@testdash#1\ifgmu@dash{%
3108   \gmu@ifnodash#1-\@@nil
3109   \gmu@dashfalse
3110   \else
3111     \gmu@dashtrue
3112   \fi
3113   \ifgmu@dash}

```

A word of explanation to the above pair of macros. `\gmu@testdash` sets `\iftrue` the `\ifgmu@dash` switch if the argument contains an explicit `-`. To learn it, an auxiliary `\gmu@ifdash` macro is used that expands to an open (un`\fied`) `\ifx` that tests whether the dash put by us is the only one in the argument string. This is done by matching the parameter string that contains a dash: if the investigated sequence contains (another) dash, `#2` of `\gmu@ifdash` becomes the rest of it and the ‘guardian’ dash put by us so then it’s nonempty. Then `#2` is took as the definiens of `\@tempa` so if it was empty, `\@tempa` becomes `x` equal `\@empty`, otherwise it is `x` not.

Why don’t we use just `\gmu@ifdash`? Because we want to put this test into another `\if . . .`. A macro that doesn’t *mean* `\if . . .` wouldn’t match its `\else` nor its `\fi` while `TEX` would skip the falsified branch of the external `\if . . .` and that would result in the ‘extra `\else`’ or ‘extra `\fi`’ error.

Therefore we wrap the very test in a macro that according to its result sets an explicit Boolean switch and write this switch right after the testing macro. (Delimiting `\gmu@testdash`’s parameter with this switch is intended to bind the two which are not one because of `TEX` nical reasons only.

Warning: this pair of macros may result in ‘extra `\else`/extra `\fi`’ errors however, if `\gmu@testdash` was `\expandaftered`.

Dates for memoirs to be able to typeset them also as diaries.

```

\ifdate 3144 \newif\ifdate
          %\newcounter{dateinsection}[section]

\data 3146 \newcommand*{\data}[1]{%
3147   \ifdate\gmu@testdash#1\ifgmu@dash\datef#1\else\datefsl#1\fi\fi}

\linedate 3149 \newcommand*{\linedate}[1]{\par\ifdate\addvspace{\dateskip}%
3150   \date@line{\footnotesize\itshape\date@biway{#1}}%
3151   \nopagebreak\else%%\ifnum\arabic{dateinsection}>0\dekbigskip\fi
3152   \addvspace{\bigskipamount}%
3153   \fi}% end of \linedate.
3155 \let\dateskip\medskipamount

\date@biway 3157 \def\date@biway#1{%
3158   \gmu@testdash#1\ifgmu@dash\datef#1\else\datefsl#1\fi}

\rdate 3160 \newcommand*\rdate[1]{\let\date@line\rightline\linedate{#1}}
\ldate 3161 \newcommand*\ldate[1]{\let\date@line\leftline\linedate{#1}}
\runindate 3162 \newcommand*\runindate[1]{%
3163   \paragraph{\footnotesize\itshape\datef#1\@@nil}\stepcounter{%
dateinsection}}

```

I’m not quite positive which side I want the date to be put to so let’s let for now and we’ll be able to change it in the very documents.

```

3166 \let\thedata\ldate
\zwrobcy 3169 \DeclareRobustCommand*\zwrobcy[1]{\emph{#1}}\_% ostinato, allegro con moto,
        garden party etc., także komplement
\tytul 3172 \DeclareRobustCommand*\tytul[1]{\emph{#1}}
        Maszynopis w świecie justowanym zrobi delikatną chorągiewkę.
maszynopis 3176 \newenvironment{maszynopis}[1][\ttfamily
3177 \hyphenchar\font=45\relax% to przypisanie jest globalne do fontu.
3178 \@tempskipa=\glueexpr\rightskip+\leftskip\relax
3179 \ifdim\gluestretch\@tempskipa=\z@
3180 \tolerance900
        sprawdziło się przy tolerancji 900
3182 \advance\rightskip\by\z@_pluso,5em\relax\fi
3183 \fontdimen3\font=\z@% zabraniamy rozciągania odstępów, ale% \fontdimen4%
        \font=\z@ dopuszczamy ich skurczenie
3185 \hyphenpenalty0% żeby nie stresować TEXa: w maszynopisie ten wspinały al-
        gorytm dzielenia akapitu powinien być wyłączony, a każdy wiersz łamany
        na ostatnim dopuszczalnym miejscu przełamania.
3189 \StoreMacro\pauzacore
\pauzacore 3190 \def\pauzacore{-\rlap{\kern-0,3em}-}%
3191 }{\par}
\justified 3195 \newcommand*\justified{%
3196 \leftskip=1\leftskip% to preserve the natural length and discard stretch and
        shrink.
3198 \rightskip=1\rightskip
3199 \parfillskip=1\parfillskip
3200 \advance\parfillskip\by\osp_plus\ifil\relax
3201 \let\\\@normalcr}
        For dati under poems.
\wherncore 3206 \newcommand\wherncore[1]{%
3207 \rightline{%
3208 \parbox{0,7666\textwidth}{
3209 \leftskiposp_plus\textwidth
3210 \parfillskiposp\relax
3211 \let\\\linebreak
3212 \footnotesize_#1}}}
\whern 3214 \newcommand\whern[1]{%
3215 \vskip\whernskip
3216 \wherncore{#1}}
\whernskip 3218 \newskip\whernskip
3219 \whernskip2\baselineskip_minus_2\baselineskip\relax
\whernup 3221 \newcommand\whernup[1]{\par\wherncore{#1}}

```

### Minion and Garamond Premier kerning and ligature fixes

„Ws” nie będzie robiło długiego „s”, bo źle wygląda przy „W”

```

\Ws 3228 \DeclareRobustCommand*\Ws{W\kern-0,08em\penalty10000\hskiposp%
        \relax
3229 s\penalty10000\hskiposp\relax}

```



```

\Wz 3231 \DeclareRobustCommand*\Wz{W\kern-0,05em\penalty10000\hskiposp%
      \relax_z}
3234 \endinput

```

## Change History

- vo.74  
 \@begnamedgroup@ifcs:  
 The catcodes of \begin and \end argument(s) don't have to agree strictly anymore: an environment is properly closed if the \begin's and \end's arguments result in the same \csname, 568  
 General:  
 Added macros to make sectioning commands of mwcls and standard classes compatible. Now my sectionings allow two optionals in both worlds and with mwcls if there's only one optional, it's the title to toc and running head not just to the latter, 3234
- vo.75  
 \@ifnextac:  
 added, 424  
 \@ifnextcat:  
 \let for #1 changed to \def to allow things like \noexpand~ , 363  
 \@ifnextif:  
 \let for #1 changed to \def to allow things like \noexpand~ , 399
- vo.76  
 General:  
 A 'fixing' of \dots was rolled back since it came out they were O.K. and that was the QX encoding that prints them very tight, 3234  
 \freeze@actives:  
 added, 2318
- vo.77  
 General:  
 \afterfi & pals made two-argument as the Marcin Woliński's analogoi are. At this occasion some redundant macros of that family are deleted, 3234
- vo.78  
 General:  
 \@namelet renamed to \n@melet to solve a conflict with the beamer class. The package contents regrouped, 3234
- vo.79  
 \not@onlypreamble:
- All the actions are done in a group and therefore \xdef used instead of \edef because this command has to use \do (which is contained in the \@preamblecmds list) and \not@onlypreamble itself should be able to be let to \do, 1179
- vo.80  
 General:  
 CheckSum 1689, 0  
 \hfillneg:  
 added, 2239
- vo.81  
 \dekfracslash:  
 moved here from pmlectionis.cls, 2497  
 \ifSecondClass:  
 moved here from pmlectionis.cls, 2466
- vo.82  
 \ikern:  
 added, 2505
- vo.83  
 \~:  
 postponed to \begin{document} to avoid overwriting by a text command and made sensible to a subsequent /, 2192
- vo.84  
 General:  
 CheckSum 2684, 0
- vo.85  
 General:  
 CheckSum 2795, 0  
 fixed behaviour of too clever headings with gmdoc by adding an \ifdim test, 3234
- vo.86  
 \texttilde:  
 renamed from texttilde since the latter is one of L<sup>A</sup>T<sub>E</sub>X accents, 2200
- vo.87  
 General:  
 CheckSum 4027, 0  
 the package goes  $\varepsilon$ -T<sub>E</sub>X even more, making use of \ifdefined and the code using UTF-8 chars is wrapped in a X<sub>Y</sub>T<sub>E</sub>X-condition, 3234

vo.88	General: Checksum 4040, 0 \RestoreEnvironment: added, 1117 \storedcsname: added, 1108 \StoreEnvironment: added, 1113	Checksum 4055, 0 removed \jobnamewoe since \jobname is always without extension. \xiispace forked to \visiblespace \let to \xxt@visiblespace of xltextra if available. The documentation driver integrated with the .sty file, 3234
vo.89	General: removed obsolete adjustment of pgf for $\text{\XeTeX}$ , 3234	vo.92 \@checkend: shortened thanks to \@ifenvir, 598 \@gif: added redefinition so that now switches defined with it are \protected so they won't expand to an further expanding or unbalanced \iftrue/false in an edef, 198
vo.90	General: Checksum 4035, 0 \XeTeXthree: adjusted to the redefinition of \verb in xltextra 2008/07/29, 2017	\@ifenvir: added, 590
vo.91	General:	General: Checksum 4133, 0

## Index

Numbers written in *italic* refer to the code lines where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used. The numbers preceded with ‘p.’ are page numbers. All the numbers are hyperlinks.

\*, 452, <u>453</u> , 454, 456, 460, 461, 462, 466, <u>2192</u> , <u>2373</u>	\@currsize, 640, 641, 642, 643, 644, 645, 646, 647, 648, 649	\@include, 2539, <u>2541</u>
\+, <u>2374</u>	\@emptyify, <u>301</u> , 302, 306	\@itemdepth, 1807, 1810, 1811
\-, <u>915</u> , 2811	\@enumctr, 1793, 1794, 1800	\@itemitem, 1811, 1812
\<...>, <u>843</u>	\@enumdepth, 1789, 1792, 1793	\@minus, 1640, 1645, 1651, 1653, 1658, 1660, 1667, 1672
\@@nil, 871, 873, 875, 2247, 2532, 2542, 2603, 2608, 2611, 2612, 2617, 3103, 3108, 3163	\@fileswithoptions, 2466	\@nobreakfalse, 1536, 2294
\@M, 1537	\@firstofmany, <u>2247</u>	\@nobreaktrue, 1531
\@afterheading, <u>1530</u>	\@gif, 184, 185, <u>192</u>	\@normalcr, 3201
\@badend, 598	\@if, 207, 208, <u>211</u>	\@nx, <u>170</u> , 342, 377, 413, 440, 1008, 1013, 1047, 1084, 1085, 1169, 1170, 1177, 1178, 1197, 1212, <u>1214</u> , <u>1215</u> , 1216, 1610, 1611, 1845, 1845, <u>1992</u> , <u>2423</u> , 2840, 2847, 2957, 2958
\@begnamedgroup, <u>537</u> , 553, 561, 566	\@ifXeTeX, <u>2011</u> , 2019, 2131, 2166, 2456, 2494, 2628, 2829	\@oarg, 950, <u>952</u> , 953
\@begnamedgroup@ifcs, 562, <u>565</u>	\@ifempty, 1475, 1492, <u>2516</u>	\@oargsq, 950, 953, 968
\@car, 1878	\@ifenvir, 590, 598	\@onlypreamble, 2296
\@ccclv, 2356	\@ifl@aded, 1203	\@parg, 957, <u>959</u> , 960
\@ccclvi, 2341	\@ifncat, 370, <u>373</u> , 388	\@pargp, 957, 960, 969
\@checkend, <u>598</u>	\@ifnextac, <u>436</u>	\@parindent, 1796, 1797, 1799, 1814, 1815, 1817
\@clsextension, 2466	\@ifnextcat, <u>363</u> , 437	
\@clubpenalty, 1542	\@ifnextif, <u>399</u>	
\@currenvir, 540, 592	\@ifnextspace, <u>459</u> , 2723, 2762	
\@currenvline, 544	\@ifnif, 406, <u>409</u>	
	\@ifnotmw, <u>1358</u> , <u>1361</u> , 1525, 1630, 1717, 1772	

<code>\@pkgextension</code> , 1204	<code>\afterfififi</code> , <u>506</u>	2137, 2138, 2139,
<code>\@preamblecmds</code> , 1198,	<code>\afteriffifi</code> , <u>500</u>	2143, 2144, 2145,
1200, 1213, 1217	<code>\afteriffififi</code> , <u>505</u>	2146, 2148, 2149,
<code>\@relaxen</code> , <u>310</u> , 311, 315	<code>\afteriffiffififi</code> , <u>504</u>	2150, 2339, 2341,
<code>\@starttoc</code> , <u>2289</u>	<code>\ahyphen</code> , <u>2811</u>	2342, 2343, 2346,
<code>\@tempdima</code> , 2635, 2639,	<code>\AKA</code> , <u>2441</u>	2355, 2356, 2357, 2358
2640, 2642, 2643, 2647	<code>\all@other</code> , <u>1986</u> , 2095, 2097	<code>\cs</code> , <u>914</u> , 920, 924, 939
<code>\@tempdima*</code> , 2640, 2642	<code>\alpha</code> , 2852	<code>\cup</code> , 3000
<code>\@tempdimb</code> , 2636, 2638, 2639	<code>\AmSTeX</code> , <u>1915</u>	<code>\czas</code> , <u>2815</u>
<code>\@textsuperscript</code> ,	<code>\arg</code> , <u>965</u> , <u>966</u>	<code>\czer</code> , <u>2370</u> , 2373, 2374
2173, 2174	<code>\AtBeginDocument</code> , 781,	<code>\czerwo</code> , <u>2369</u> , 2370
<code>\@toodeep</code> , 1790, 1808	964, 1211, 1329, 2034,	<code>\dag</code> , 2374
<code>\@topnewpage</code> , 1436	2045, 2198, 2390,	<code>\data</code> , <u>3146</u>
<code>\@undefined</code> , 152	2718, 2789, 2791	<code>\date@biway</code> , 3150, <u>3157</u>
<code>\@whilenum</code> , 2341, 2356	<code>\begin</code> , <u>561</u>	<code>\date@line</code> , 3150, 3160, 3161
<code>\@xa</code> , <u>169</u> , 193, 195, 212, 213,	<code>\begin*</code> , <u>561</u>	<code>\datef</code> , <u>3021</u> , <u>3061</u> , 3147,
283, 293, 298, 329,	<code>\bgcolor</code> , 2992	3158, 3163
330, 388, 592, 593,	<code>\BibTeX</code> , <u>1918</u>	<code>\datefsl</code> , <u>3039</u> , <u>3080</u> ,
656, 816, 939, 1001,	<code>\Biggl</code> , 2943	3147, 3158
1008, 1013, 1078,	<code>\biggl</code> , 2941	<code>\dateskip</code> , 3149, 3155
1084, 1085, 1138,	<code>\Biggr</code> , 2944	<code>\deadcycles</code> , 2572
1158, 1169, 1170,	<code>\biggr</code> , 2942	<code>\DeclareLogo</code> , <u>1834</u> , <u>1851</u> ,
1177, 1178, 1508,	<code>\Bigl</code> , 2939	1877, 1888, <u>1918</u> ,
1556, 1608, 1739,	<code>\bigl</code> , 2937	1924, 1927, 1929,
1751, 1794, 1812,	<code>\Bigr</code> , 2940	1932, 1935, 1939,
1840, 1986, 1992,	<code>\bigr</code> , 2938	1941, 1944, 1951
2001, 2291, 2384,	<code>\bigskipamount</code> , 2236, 3152	<code>\DeclareRobustCommand</code> ,
2600, 2665, 2668,	<code>\bnamegroup</code> , <u>553</u>	2616
2840, 2847, <u>2957</u> , 2957	<code>\boldmath</code> , 1878	<code>\DeclareRobustCommand*</code> ,
<code>\@xifncat</code> , 375, <u>388</u>	<code>\box</code> , 1899, 2936	636, 673, 674, 675,
<code>\@xifnif</code> , 411	<code>\bslash</code> , <u>760</u> , 914, 1008,	676, 677, 678, 807,
<code>\acro</code> , <u>2405</u> , 2440, 2441, 2445	1085, 1109, 1499,	881, 888, 903, 914,
<code>\acrocore</code> , 2425, <u>2435</u>	1500, 1501, 2758,	920, 924, 1335, <u>1848</u> ,
<code>\adashes</code> , <u>2789</u> , 2789,	2759, 2768, 2769	2047, 2172, 2192,
<u>2791</u> , 2797	<code>\bullet</code> , 2829	2200, 2405, 2443,
<code>\addfontfeature</code> , 2048,	<code>\c@gm@PronounGender</code> , <u>1257</u>	2719, 2734, 2747,
2078, 2080, 2103,	<code>\c@NoNumSecs</code> , <u>1331</u>	2756, 2766, 3169,
2168, 2443, 2654,	<code>\c@secnumdepth</code> , 1385	3172, 3228, 3231
2988, 3001	<code>\chardef</code> , 2829	<code>\DeclareTextCommand</code> , 1845
<code>\addto@macro</code> , <u>282</u> , 289	<code>\cite</code> , 1228	<code>\DeclareTextCommandDefault</code> ,
<code>\addtoheading</code> , <u>1506</u>	<code>\ClassError</code> , 1498	1847
<code>\addtomacro</code> , 289, 2972	<code>\cleardoublepage</code> , 1345	<code>\defobeylines</code> , <u>2224</u>
<code>\addtonummacro</code> , <u>1291</u>	<code>\clubpenalty</code> , 1537, 1542	<code>\dekbigskip</code> , 2236
<code>\addtotoks</code> , <u>297</u>	<code>\cmd</code> , <u>939</u>	<code>\dekfracc</code> , <u>2093</u> , <u>2102</u> , 2492
<code>\AE</code> , 2622	<code>\cmd@to@cs</code> , 939, <u>941</u>	<code>\dekfraccsimple</code> , <u>2491</u> , 2497
<code>\afterfi</code> , 440, 463, 464,	<code>\color</code> , 2369	<code>\dekfraccslash</code> , 2486,
497, 566, 567, 594,	<code>\continuum</code> , 2400	<u>2494</u> , <u>2495</u>
595, 861, 895, 966,	<code>\copy</code> , 1861, 1891, 1905,	<code>\dekmedskip</code> , <u>2235</u>
1000, 1077, 2014,	2991, 3003	<code>\deksmallskip</code> , <u>2233</u>
2414, 2430, 2518,	<code>\count</code> , 1292, 1293, 1294,	<code>\Delta</code> , 2853
2519, 2580, 2608,	1866, 1867, 1868,	<code>\dimen</code> , 2135, 2138, 2142,
2722, 2761	1869, 1870, 1871,	2145, 2304, 2305,
<code>\afterfifi</code> , <u>499</u> , 1055,	1872, 1873, 2136,	2306, 2686, 2687
1060, 2013, 2808		

\dimeexpr, 2877, 2882, 2888, 2935, 2992	\FormatRunInHeading, 1668, 1673	\gma@quantifierhook, 2886, 2889, 2967, 2969, 3001, 3007
\discre, <u>854</u> , 864, 883	\freeze@actives, <u>2338</u>	\gma@tempa, 2875, 2877, 2880, 2882, 2931, 2935, 2956, 2959
\discret, <u>856</u> , 861	\g@emptyify, <u>306</u> , 307	\gma@tempb, 2932, 2935
\divide, 1868, 1871, 1872, 2137, 2139, 2144, 2146, 2638, 2639, 2644	\g@relaxen, <u>315</u> , 316	\gm@math, 2839, 2975, 2979
\dp, 2920, 2932, 2935, 2992	\gaddtomacro, <u>277</u>	\gm@mathhook, <u>2972</u>
\dywiz, 2808	\garamath, <u>2987</u>	\gml@StoreCS, 1034, 1057, 1095
	\gemptify, <u>307</u> , 307	\gml@storemacros, 1035, 1046, 1055, 1060, 1098
\edef@other, 2000, 2005	\glet, <u>269</u> , 1557	\gmobeyspaces, <u>2216</u>
\eequals, <u>2318</u>	\glueexpr, 3178	\gmshowlists, <u>326</u>
\egRestore@Macro, 1070, 1072	\gluestretch, 3179	\GMtextsuperscript, <u>2165</u>
\egRestore@MacroSt, 1070, <u>1073</u>	\gm@atppron, <u>1259</u> , 1263, 1264, 1265, 1266, 1268, 1269, 1270, 1271	\gmu@acroinner, 2411, 2421, 2422, 2430
\egroupfirstofone, 694	\gm@clearpagesduetoopenright, <u>1344</u> , 1404	\gmu@acrospace, 2405, 2410, 2410, 2414
\egStore@Macro, 990, <u>995</u>	\gm@dontnumbersectionsoutofmainmatter, <u>1342</u> , 1388	\gmu@checkaftersec, 1551, 1610
\egStore@MacroSt, 990, <u>996</u>	\gm@duppa, <u>2094</u> , 2095, 2097	\gmu@copyright, 2452, 2453
\emdash, <u>2774</u>	\gm@gobmacro, 1986, <u>1992</u>	\gmu@dashfalse, 3109
\emptyify, <u>302</u> , 302, 2050, 2967, 2971	\gm@hyperrefstepcounter, <u>1334</u> , <u>1337</u> , 1417	\gmu@dashtrue, 3111
\enamegroup, <u>555</u>	\gm@ifnac, 437, <u>439</u>	\gmu@def, 1360, <u>1362</u> , 1362, 1381
\endlist, 1803, 1820	\gm@letspace, 455, 463	\gmu@dekfracc, <u>2077</u> , 2096, 2098
\englishdate, <u>3060</u>	\gm@notprerr, 1209, 1216	\gmu@dekfraccsimple, 2098, <u>2484</u> , 2492
\enoughpage, <u>2302</u>	gm@PronounGender, <u>1257</u>	\gmu@denominatorkern, 2079, 2108, 2486
\ensuremath, 815, 828, 1936, 2175	\gm@pswords, 871, <u>873</u> , 875	\gmu@discretionaryslash, <u>883</u> , 894
enumerate*, <u>1788</u>	\gm@sec, <u>1755</u> , 1766, 1767	\gmu@dywiz, 2807, 2811
\env, <u>920</u>	\gm@secini, <u>1720</u> , 1739, <u>1745</u> , 1751, 1764	\gmu@fileext, <u>2534</u> , <u>2543</u> , 2543, 2561
\equals, <u>2316</u>	\gm@secmarkh, 1747	\gmu@filename, <u>2533</u> , 2546, 2558, 2561, 2564, 2573
\eTeX, 1935, 1939	\gm@secstar, 1723, 1733, 1740, 1752, <u>1766</u> , <u>1767</u>	\gmu@getaddvs, <u>1602</u> , 1602, 1608
\everypar, 1533, 1543, 2708	\gm@secx, 1755, <u>1756</u>	\gmu@gettext, <u>2532</u> , 2542
\exists, 2889	\gm@secxx, <u>1722</u> , <u>1750</u> , 1757	\gmu@ifnodash, <u>3103</u> , 3108
	\gm@straightensec, <u>1761</u> , 1770	\gmu@LastSkip, 2824, 2825, 2827
\f@encoding, 2353	\gm@targetheading, <u>1335</u> , <u>1338</u>	\gmu@luzniej, <u>2691</u> , 2694, 2696
\f@series, 1878	\gma, <u>2980</u>	\gmu@nl@reserveda, 1168, 1171, 1176, 1179
\fakern, <u>2961</u>	\gma@arrowdash, <u>2990</u> , 2999, 3005, 3006	\gmu@numeratorkern, 2078, 2107, 2108, 2485
\fakesc@extrascap, 2642, 2657	\gma@bare, 2976, 2978	
\fakesc@scaps, <u>2616</u>	\gma@bracket, 2978, <u>2979</u>	
\fakesc@score, 2594, 2619	\gma@checkbracket, <u>2977</u> , 2981	
\fakesc@extrascap, <u>2657</u>	\gma@dollar, <u>2975</u> , 2976, 2981	
\file, <u>888</u>	\gma@gmathhook, 2964, 2971, 2972, 2994	
\fixcopyright, <u>2455</u>		
\fontencoding, 2447		
\fooatletter, <u>698</u>		
\foone, 694, 698, 717, 720, 728, 744, 764, 768, 771, 923, 2215, 2223, 2788, 2810, 2903		
\forall, 2886		
\FormatHangHeading, 1639, 1646, 1654, 1661		

\gmu@prevsec, 1532, 1534, 1556, 1563, 1593	\gmu@xfraccdef, 2070, 2084, 2085, 2086, 2087, 2088, 2089, 2090, 2091, 2092	\ifodd, 1261
\gmu@printslashes, 888, 890, 890, 892, 895	\gn@melet, 1175	\ifSecondClass, 2464
\gmu@resa, 2382, 2384	\gobble, 705, 706, 2957	\ikern, 2505
\gmu@reserveda, 452, 454, 460, 462, 592, 594, 1047, 1050, 1053, 1508, 1510, 1557, 1558, 1561, 1840, 1842, 1844, 1845, 1846, 1991, 1993, 2000, 2001, 2005, 2006, 2517, 2518	\gobbletwo, 707	\IMO, 2440
\gmu@reservedb, 593, 594	\grefstepcounter, 236, 252	\in, 3007
\gmu@RPif, 2368, 2380, 2391, 2400	\grelaxen, 316, 316, 1534	\inclasthook, 2562, 2584
\gmu@scalar, 2629, 2633, 2634, 2640	\hathat, 924	\includegraphics, 2154
\gmu@scalematchX, 2619, 2631, 2655	\HeadingNumber, 1414, 1416	\infty, 2864
\gmu@scapLetters, 2590, 2600, 2605	\HeadingNumberedfalse, 1343, 1385	\itemindent, 1796, 1814
\gmu@scapSpaces, 2603, 2608, 2612	\HeadingRHeadText, 1398	itemize*, 1806
\gmu@scapss, 2611, 2617	\HeadingText, 1400	\iteracro, 2404, 2408
\gmu@scscale, 2648, 2654	\HeadingTOCText, 1399	\justified, 3195
\gmu@setheading, 1607, 1613, 1614	\HeShe, 1268	\labelsep, 1798, 1816
\gmu@setsetSMglobal, 1033, 1038, 1094	\heshe, 1263	\labelwidth, 1797, 1798, 1815, 1816
\gmu@setSMglobal, 1040, 1042, 1060	\hfillneg, 2239	\larger, p. 9, 673, 2876, 2881, 2937, 2938, 2941, 2942, 2943, 2944
\gmu@SMdo@scope, 1137, 1139, 1142, 1143, 1157	\hgrefstepcounter, 251	\largerr, p. 9, 677, 2939, 2940
\gmu@SMdo@setscope, 1135, 1141, 1155	\HimHer, 1270	\LaTeXe, 1830, 1877
\gmu@SMglobalfalse, 1002, 1015, 1042, 1051, 1079, 1088, 1145	\himher, 1265	\LaTeXpar, 1888
\gmu@SMglobaltrue, 978, 1040	\HisHer, 1269	\ldate, 3161, 3166
\gmu@smtempa, 1006, 1014, 1082, 1087	\hisher, 1264	\leftarrow, 2896, 3005
\gmu@tempa, 2664, 2666, 2667, 2668	\HisHers, 1271	\leftline, 3161
\gmu@testdash, 3107, 3147, 3158	\hishers, 1266	\leftmargin, 1795, 1813
\gmu@tilde, 2187, 2192, 2203	\hrule, 1581	\leftrightharrow, 2898
\gmu@whonly, 2579, 2580	\hunskip, 259	\linebreak, 3211
\gmu@xedekfracplain, 2055, 2101	\hyphenpenalty, 875, 2725, 3185	\linedate, 3149, 3160, 3161
\gmu@xedekfracstar, 2055, 2069	\if@afterindent, 1538	\list, 1794, 1812
	\if@filesw, 2291, 2545, 2557, 2565	\listparindent, 1799, 1817
	\if@mainmatter, 1343	\liturgiques, 2367
	\if@nobreak, 1535	\longpauza, 2777, 2778
	\if@openright, 1345	\looseness, 2697, 2708
	\if@specialpage, 1403	\lpauza, 2747
	\if@twoside, 1429	luzniej, 2701
	\ifcurname, 566	luzniej*, 2706
	\ifdate, 3144, 3147, 3149	\luzniejcore, 2693, 2701
	\ifdefined, 161, 723, 782, 2012, 2717, 2787	
	\iffontchar, 2071	\macro, 1992
	\ifgmu@dash, 3101, 3107, 3113, 3147, 3158	\MakeUppercase, 2594
	\ifgmu@postsec, 1553, 1592, 1600	\mapsto, 2998
	\ifgmu@SMglobal, 976, 1000, 1007, 1039, 1077, 1083, 1142	\marg, 945, 969
	\ifHeadingNumbered, 1384, 1412	maszynopis, 3176
		\math@arg, 965, 966
		\mathbin, 2865, 2869, 2894, 2895, 2924, 2925, 2954, 2955, 3000, 3007
		\mathchoice, 2873, 2892, 2904, 2946, 2996
		\mathclose, 2922, 2923, 2938, 2940, 2942, 2944, 2952

<code>\mathfrak</code> , 2400	<code>\napapierkicore</code> , 2681, 2685	<code>\PlainTeX</code> , 1927
<code>\mathit</code> , 2840, 2851	<code>\napapierkistretch</code> , 2679, 2682	<code>\pm</code> , 2868, 2869
<code>\mathop</code> , 2873	<code>\nawj</code> , 2710	<code>\polskadata</code> , 3020, 3057
<code>\mathopen</code> , 2909, 2923, 2937, 2939, 2941, 2943, 2951	<code>\nazwired</code> , 2834	<code>\possfil</code> , 929
<code>\mathpunct</code> , 2908	<code>\neg</code> , 2865, 2960	<code>\ppauza</code> , 2766
<code>\mathrel</code> , 2866, 2870, 2896, 2897, 2898, 2924, 2925, 2926, 2958, 2999, 3005, 3006	<code>\neq</code> , 2866, 2955	<code>\prependtomacro</code> , 292
<code>\mathrm</code> , 2847, 2853, 2859, 2865, 2866, 2868, 2869, 2870, 2884	<code>\neqb</code> , 2955	<code>\printspaces</code> , 871, 881
<code>\Mathstrutbox@</code> , 2920	<code>\newcount</code> , 2691	<code>\protected</code> , 181, 193, 198, 212, 236, 251, 259
<code>\medmskip</code> , 861	<code>\newcounter</code> , 1257, 1331	<code>\przeniesvskip</code> , 2823
<code>\meta</code> , 807, 843, 945, 952, 959	<code>\newgif</code> , 181	<code>\psi</code> , 2862
<code>\meta@font@select</code> , 818, 837	<code>\newskip</code> , 3218	<code>\quad</code> , 2834
<code>\mkern</code> , 2961	<code>\newwrite</code> , 2291	<code>\quantifierhook</code> , 2968, 2988
<code>\mskip</code> , 861	<code>\nfss@text</code> , 816	<code>\qxcopyright</code> , 2450
<code>\multiply</code> , 1867, 1870, 2646, 2696, 2707	<code>\nieczer</code> , 2375	<code>\qxcopyrights</code> , 2451, 2456
<code>\mw@getflags</code> , 1556	<code>\nobreakspace</code> , 2611	<code>\qxenc</code> , 2447, 2450, 2453
<code>\mw@HeadingBreakAfter</code> , 1405, 1425, 1440, 1444, 1474, 1557	<code>\nocite</code> , 1227	
<code>\mw@HeadingBreakBefore</code> , 1402, 1473, 1558	<code>\nohy</code> , 2509	<code>\rdate</code> , 3160
<code>\mw@HeadingLevel</code> , 1382, 1385	<code>\nolimits</code> , 2888, 2889	<code>\real</code> , 2640, 2642
<code>\mw@HeadingRunIn</code> , 1420, 1473	<code>NoNumSecs</code> , 1331	<code>\reflectbox</code> , 1947, 1954
<code>\mw@HeadingType</code> , 1401, 1532, 1564, 1565, 1578	<code>\not@onlypreamble</code> , 1196, 1200, 1201, 1202, 1203, 1204	<code>\relaxen</code> , 311, 311, 1470, 2797, 2850
<code>\mw@HeadingWholeWidth</code> , 1423, 1474	<code>\nu</code> , 2856	<code>\relsize</code> , p. 9, 636, 637, 673, 674, 675, 676, 677, 678
<code>\mw@normalheading</code> , 1427, 1436, 1439, 1443, 1613	<code>\numexpr</code> , 2816, 2817	<code>\renewcommand*</code> , 2391
<code>\mw@processflags</code> , 1475	<code>\nummacro</code> , 1280	<code>\RequirePackage</code> , 2026, 2288, 2384, 2624
<code>\mw@runinheading</code> , 1421, 1614	<code>\oarg</code> , 950	<code>\resetMathstrut@</code> , 2914
<code>\mw@secdef</code> , 1480, 1481, 1482, 1488	<code>\oldLaTeX</code> , 1829	<code>\resizebox</code> , 2153
<code>\mw@section</code> , 1479	<code>\oldLaTeXe</code> , 1830	<code>\resizegraphics</code> , 2132, 2152
<code>\mw@sectionxx</code> , 1381	<code>\omega</code> , 2863	<code>\Restore@Macro</code> , 1072, 1075, 1095, 1105
<code>\mw@secundef</code> , 1484, 1493, 1496	<code>\OnAtLine</code> , 2312	<code>\Restore@Macros</code> , 1091, 1093
<code>\mw@setflags</code> , 1485	<code>\PackageError</code> , 1215, 2757, 2767	<code>\Restore@MacroSt</code> , 1073, 1081
<code>\n@melet</code> , 1167, 1507, 1511, 1724, 1730, 1764, 2074	<code>\PackageWarning</code> , 664, 668	<code>\RestoreEnvironment</code> , 1117
<code>\nameshow</code> , 329	<code>\pagebreak</code> , 1428, 1440, 1444	<code>\RestoreMacro</code> , 1069, 1362, 2033, 2035
<code>\nameshowthe</code> , 330	<code>\pagegoal</code> , 2304	<code>\RestoreMacro*</code> , 1119, 2035
<code>\napapierki</code> , 2684	<code>\pagetotal</code> , 2305	<code>\RestoreMacros</code> , 1091
	<code>\paragraph</code> , 3163	<code>\RestoringDo</code> , 1154
	<code>\ParanoidPostsec</code> , 1589	<code>\rightarrow</code> , 2897, 3006
	<code>\parg</code> , 957	<code>\rightline</code> , 3160, 3207
	<code>\partial</code> , 2867	<code>\romannumeral</code> , 1793, 1811
	<code>\partopsep</code> , 1795, 1813	<code>\rotatebox</code> , 2886, 2889, 2894, 2895
	<code>\pauza</code> , 2734	<code>\rs@size@warning</code> , 655, 660, 664
	<code>\pauzacore</code> , 2724, 2725, 2730, 2738, 2743, 2748, 2777, 2780, 3189, 3190	<code>\rs@unknown@warning</code> , 650, 668
	<code>\pdfTeX</code> , 1939	<code>\runindate</code> , 3162
	<code>\pdfTeX</code> , 1941	
	<code>\Phi</code> , 2859	
	<code>\phi</code> , 2858	
	<code>\pi</code> , 2857	
	<code>\pk</code> , 903	

<code>\scantokens</code> , 2902	<code>\TeXbook</code> , 1932, 1933	<code>\vs</code> , 864, 871, 875
<code>\scshape</code> , 1925, 2435, 3000	<code>\textbullet</code> , 2829, 2829	<code>\vspace*</code> , 2826
<code>\secondclass</code> , 2463	<code>\textcolor</code> , 2375	
<code>\SecondClasstrue</code> , 2465	<code>\textlarger</code> , 675	<code>\wd</code> , 1859, 1862, 1892, 1897,
<code>\sectionsign</code> , 2844	<code>\textsl</code> , 1932	1905, 1906, 2136,
<code>\SetSectionFormatting</code> ,	<code>\textsmaller</code> , 676	2888, 2991, 2992,
1470, 1471, 1632,	<code>\textstyle</code> , 1879	3003, 3004
1636, 1644, 1652,	<code>\textsuperscript</code> , 2167,	<code>\Web</code> , 1929
1659, 1666, 1671	2172	<code>\wedge</code> , 2895, 2960
<code>\SetTwoheadSkip</code> , 1616,	<code>\texttilde</code> , 2200	<code>\whenonly</code> , 2578
1643, 1651, 1658	<code>\textwidth</code> , 3208, 3209	<code>\whern</code> , 3214
<code>\sfname</code> , 881, 891	<code>\thedata</code> , 3166	<code>\wherncore</code> , 3206, 3216, 3221
<code>\shortpauza</code> , 2779	<code>\thickmuskip</code> , 2962	<code>\whernskip</code> , 3215, 3218, 3219
<code>\showboxbreadth</code> , 326	<code>\thr@@</code> , 1789, 1807	<code>\whernup</code> , 3221
<code>\showboxdepth</code> , 326	<code>\time</code> , 2816, 2817	<code>\WPheadings</code> , 1631
<code>\ShowFont</code> , 2352	<code>\tinycae</code> , 2622	<code>\Ws</code> , 3228
<code>\showlists</code> , 326	<code>\TODO</code> , 2251	<code>\wyzejnizej</code> , 2663
<code>\showthe</code> , 330	<code>\toks</code> , 1508, 1509, 1510,	<code>\Wz</code> , 3231
<code>\sigma</code> , 2860	1603, 1604, 1610, 1611	
<code>\sim</code> , 2870	<code>\tolerance</code> , 2696, 2707, 3180	<code>\Xedekfrac</code> , 2055
<code>\SliTeX</code> , 1924	<code>\tOnLine</code> , 2310	<code>\XeLaTeX</code> , 1951
<code>\smaller</code> , <i>p.</i> 9, 674, 2485, 2488	<code>\truetextsuperscript</code> ,	<code>\XeTeX</code> , 1944
<code>\smallerr</code> , <i>p.</i> 9, 678, 2655	2169, 2171	<code>\XeTeXinputencoding</code> , 162
<code>\smallskipamount</code> , 2233,	<code>\twocoltoc</code> , 2287, 2296	<code>\XeTeXpicfile</code> , 2133, 2150
2234	<code>\tytul</code> , 3172	<code>\XeTeXthree</code> , 2017
<code>\smartunder</code> , 746	tytulowa, 2831	<code>\XeTeXversion</code> , 151, 152,
<code>\SMglobal</code> , 978		161, 723, 2012, 2013,
<code>\stepnummacro</code> , 1282	<code>\udigits</code> , 2047, 2050,	2717, 2787
<code>\Store@Macro</code> , 995, 998, 1034	2485, 2488	<code>\xiand</code> , 769
<code>\Store@Macros</code> , 1031, 1032	<code>\undeksmallskip</code> , 2234	<code>\xiibackslash</code> , 756, 760
<code>\Store@MacroSt</code> , 996, 1005	<code>\unex@namedef</code> , 1999	<code>\xiilbrace</code> , 731
<code>\Stored@Macro</code> , 1104, 1105	<code>\unex@nameuse</code> , 2004	<code>\xiipercen</code> , 765
<code>\storedcsname</code> , 1108	<code>\unexpanded</code> , 293	<code>\xiirbrace</code> , 732
<code>\StoredMacro</code> , 1104	<code>\unless</code> , 2013	<code>\xiispace</code> , 342, 343, 772, 785
<code>\StoreEnvironment</code> , 1113	<code>\usc</code> , 2443, 2445	<code>\xiistring</code> , 341
<code>\StoreMacro</code> , 989, 1362,	<code>\uscacro</code> , 2445	<code>\xiiunder</code> , 721, 724, 725
1881, 2025, 3189	<code>\usecounter</code> , 1800	<code>\xxt@visible</code> , 782, 783
<code>\StoreMacro*</code> , 1115, 1882	<code>\value</code> , 1261	
<code>\StoreMacros</code> , 1031, 2950	<code>\varepsilon</code> , 1879, 1936, 2854	<code>\yesy</code> , 2510
<code>\StoringAndRelaxingDo</code> ,	<code>\varnothing</code> , 3001	
1134	<code>\varsigma</code> , 2861	<code>\zf@scale</code> , 2630, 2633, 2634
<code>\subs</code> , 718, 748	<code>\vartheta</code> , 2855	<code>\zwrobcy</code> , 3169
<code>\sum</code> , 2885	<code>\vee</code> , 2894, 2960	
	<code>\verb</code> , 2025, 2033	<code>\-</code> , 2756, 2793
<code>\TB</code> , 1933	<code>\visible</code> , 783, 785, 864	<code>\-</code> , 2719, 2774, 2792