

Grzegorz Murzynowski

The gmverb Package^{*}

Copyright © 2005, 2006, 2007, 2008, 2009, 2010

by Grzegorz ‘Natrór’ Murzynowski

natror (at) o2 (dot) pl

This program is subject to the L^AT_EX Project Public License.

See <http://www.ctan.org/tex-->

[archive/help/Catalogue/licenses.lpp1.html](http://www.ctan.org/tex--archive/help/Catalogue/licenses.lpp1.html) for the details of that license.

LPPL status: “author-maintained”.

For the documentation please refer to the file(s)

gmverb.{gmd,pdf}.

```
49 \NeedsTeXFormat{LaTeX2e}
50 \ProvidesPackage{gmverb}
51   [2010/09/07_v0.95_After_shortvrb (FM) _but_my_way (GM) ]
54 <★master>
    (A handful of meta-settings skipped)
83 </ master>
```

Contents

Intro, usage	1	Almost-Knuthian \ttverbatim	6
The package options	4	The core: from shortvrb	7
Installation	4	doc- and shortvrb-compatibility	14
Contents of the gmverb.zip archive	4	Grey visible spaces	14
Compiling of the documentation	4	Verbatim specials—CSes in verbatims	15
The code	5	Partial \verb in arguments	17
Preliminaries	5	Change History	18
The breakables	5		

Intro, usage

This package redefines the `\verb` command and the `verbatim` environment so that the verbatim text can break into lines, with % (or another character chosen to be the comment char) as a ‘hyphen’. Moreover, it allows the user to define their own `verbatim`-like environments provided their contents would be not *horribly* long (as long as a macro’s argument may be at most).

^{*} This file has version number dated .

This package also allows the user to declare a chosen char(s) as a ‘short verb’ e.g., to write `| \a\verbatim\example |` instead of `\verb| \a\verbatim\example |`.

The `gmverb` package redefines the `\verb` command and the `verbatim` environment in such a way that `|`, `{` and `\` are breakable, the first with no ‘hyphen’ and the other two with the comment char as a hyphen. I.e. `{ <subsequent text> }` breaks into `{ %`

`<subsequent text> }` and `<text> \mymacro` breaks into `<text> % \mymacro`.

`\nobreakbslash` (If you don’t like line breaking at backslash, there’s the `\nobreakbslash` declaration (observing the common scoping rules, hence OCSR) and an analogous declaration for the left brace: `\nobreaklbrace`.)

`\VerbHyphen` The default ‘hyphen’ is `%` since it’s the default comment char. If you wish another char to appear at the line break, use the `\VerbHyphen` declaration that takes `<char>` as the only argument. This declaration is always global.

`\verbeolOK` Another difference is the `\verbeolOK` declaration (OCSR). Within its scope, `\verb` allows an end of a line in its argument and typesets it just as a space.

As in the standard version(s), the plain `\verb` typesets the spaces blank and `\verb*` makes them visible.

`\MakeShortVerb` Moreover, `gmverb` provides the `\MakeShortVerb` macro that takes a one-char control sequence as the only argument and turns the char used into a short verbatim delimiter, e.g., after `\MakeShortVerb* \` (as you guess, the declaration has its starred version, which is for visible spaces, and the non-starred for the spaces blank) you may type `| \mymacro |` to get `\mymacro` instead of typing `\verb+ \mymacro +`. Because the char used in this example is my favourite and used just this way by DEK in the *The T_EX book*’s format, `gmverb` provides a macro `\dekclubs` as a shorthand for `\MakeShortVerb (*) % \`.

`\DeleteShortVerb` Be careful because such active chars may interfere with other things, e.g. `|` with the `tikz` package. If this happens, you can declare `\DeleteShortVerb \` and the previous meaning of the char used shall be restored.

`\OldMakeShortVerb` One more difference between `gmverb` and `shortverb` is that the chars `\active`ated by `\MakeShortVerb` in the math mode behave as if they were ‘other’, so you may type e.g., `$2|o$` to get `2|o` and `+ \active`ated this way is in the math mode typeset properly etc. However, if you don’t like such a conditional behaviour, you may use `\Old \MakeShortVerb` instead, what I do when I like to display short verbatims in `display-math`.

`\dekclubs` There’s one more declaration provided by `gmverb`: `\dekclubs`, which is a shorthand for `\MakeShortVerb \`, `\dekclubs*` for `\MakeShortVerb* \` and `\olddekclubs` for `\OldMakeShortVerb \`.

`\edverbs` There’s one more declaration, `\edverbs` that makes `\[` checks if the next token is an active char and opens an `\hbox` if so. That is done so that you can write (in `\edverbs`’ and `\dekclubs`’ scope)

`\[<verbatim stuff> | \]`

instead of

`\[\hbox{ | <verbatim stuff> | } \]`

to get a displayed shortverb.

Both versions of `\dekclubs` OCSR.

The `verbatim` environment inserts `\topsep` before and after itself, just as in standard version (as if it was a `list`).

In August 2008 Will Robertson suggested grey visible spaces for `gmdoc`. I added a respective option to `gmdoc` but I find them so nice that I want to make them available for all verbatim environments so I bring here the declaration `\VisSpacesGrey`. It redefines

`\VisSpacesGrey`

visspacesgrey

only the visible spaces so affects `\verb*` and `verbatim*` and not the unstarred versions. The colour of the visible spaces is named `visspacesgrey` and you can redefine it `xcolor` way.

`\verbatimspecials`

We also provide the `\verbatimspecials` declaration that takes six arguments:

- #1 m a char for verbatim escape char (for catcode 0), has to be unbraced¹,
- #2 m a char for group starter (for catcode 1), has to be unbraced,
- #3 m a char for group ender (for catcode 2), has to be unbraced,
- [#4] (optional) a char for verbatim math shift (for catcode 3); it has to be in square brackets if present. If absent, nothing is set for the verbatim math shift,
- [#5] (optional) a char for the shorthand for `\metachar`; it has to be in square brackets if present. If provided, e.g., `>` as I suggest in `\GMverbatimspecials`, then it itself becomes an active char let-equal to `\meta`, and a CS made of it, `\>` in this example, becomes `\string`.
- {#6} b optional in curly braces, additional stuff (commands) to be executed in a verbatim.

All the specials defined this way, except the meta char, if preceded with the escape char, will be typeset verbatim.

For example, after telling \TeX

```
\verbatimspecials</>«»_[]_>{\def\|{\$vert$}}
```

(the slash is Unicode Fractional Slash, spaces are ignored) you can write

```
|\macro<arg«arg. &n+1&»\>[No >]Value>(T|F>)|
```

to get

```
\macro{<arg. n + 1>}\[No]Value(T|F)
```

Note also that `<|` is a control sequence so it doesn't delimit the short verbatim `|'s` argument.

The `\verbatimspecials` declaration OCSR. Subsequent uses of it override the previous settings. If you specified the optionals at first and then specify `\verbatimspecials` without optionals, the previous optional settings are forgotten.

`\noververbatimspecials`

To turn the 'verbatim specials' off write `\noververbatimspecials`, which OCSR too.

`\sup`

Note that although we don't provide a 'verbatim superscript' nor 'verbatim subscript', you have the `\sup` and `\sub` CS'es defined by `gmutils`.

`\sub`

The 4th argument for the math shift is optional because you can use \TeX 's `\(` and `\)`.

`\(`

`alltt`

The `\verbatimspecials` declaration goes a step further than \TeX 's `alltt` and Til Tantau's beamer's `semiverbatim` environments. To get their effect, declare

`semiverbatim`

```
\verbatimspecials\{\}
```

`\scanverb`

There is something for verbatims in arguments of commands: `\scanverb[*]{<text>}`. However there are some limitations: if `%` is the comment char (which is usual situation), then you cannot use `%` in `<text>`, or rather, `%` will act as comment char anyway. Moreover, spaces are ignored. This last limitation may be worked around if you declare `\verbatimspecials`, say `</` (fraction slash) as the escape char. Then you can use `<_` to put a space which will be typeset blank in the unstarred version and visible with star.

Not so long ago I started to use the 'broken bar' (U+00A6, `|`) character as a hyphen in hyperlinks, because it seems not to occur in hyperlinks at all unlike hyphen. I suggest the same char for verbatims, but if you don't like it, there's the `\verbDiscretionary|`

`verbDiscretionaryHyphen`

¹ To be precise, the arguments cannot be wrapped in curly braces because those are recatcoded to 'other'. But if you make some other pair of chars category 1 and 2 that are not on the `\dospecials` list, then you can wrap the arguments in those chars, but what for?

Hyphendeclaration that takes two arguments. Broken bar is declared as

```
\verbDiscretionaryHyphen{"A6}{|}
```

Since version 0.95 (August 2010) this package also provides the `\VerbatimPitch` declaration that modifies the `verbatim` environments and derivatives so that the environment contents (recatcoded, i.e. “sanitised”) are wrapped in the macro `\Verbatim|Contents` and therefore available after environment’s end (after `\endverbatim` to be precise, so also in the end-def of a derivative `verbatim`).

This may be useful for T_EXnical examples: you can rescan the contents of a `verbatim` with `\scantokens` and execute/typeset it. Such a thing is done in the `gmdoc` package, see the `verbatim@p` environment.

The package options

As many good packages, this also does not support any options.

Installation

Unpack the `\jobname-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_EX distribution’s files’ database most probably.

Contents of the gmverb.zip archive

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

```
gmverb.sty
README
gmverb.pdf
gmverb.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^AT_EX on the `gmverb.sty` file twice (`xelatex gmverb.sty` in the directory you wish the documentation to be in, you don’t have copy the `.\gmdExt` file there, T_EX will find it), then `MakeIndex` on the `\jobname.idx` file, and then L^AT_EX on `\jobname.\gmdExt` once more.

`MakeIndex` shell commands:

```
makeindex -r gmverb
makeindex -r -s gmglo.ist -o gmverb.gls gmverb.glo
```

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 `gm-docc.cls`), `gmverb.sty`, the `gmutils` bundle, `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.

Moreover, you should put the `gmglo.ist` file, a `MakeIndex` style for the changes’ history, into some `texmf/makeindex (sub)directory`.

Then you should refresh your T_EX distribution’s files’ database most probably.

If you had not installed the mwcls classes (available on CTAN and present in T_EX 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.

The code

Preliminaries

```
435 \RequirePackage{gmcommand}[2010/06/20]
```

For \firstofone, \afterfi, \gmobeyspaces, \@ifnextcat, \foone and \noexpand's and \expandafter's shorthands \@nx and \@xa resp. and \DeclareCommand.

Someone may want to use another char for comment, but we assume here 'ortho-doxy'. Other assumptions in gmdoc are made. The 'knowledge' what char is the comment char is used to put proper 'hyphen' when a verbatim line is broken.

```
\verbhyphen 449 \let\verbhyphen\xiipercen
```

Provide a declaration for easy changing it. Its argument should be of \<char> form (a <char>₁₂ is also allowed).

```
\VerbHyphen 455 \def\VerbHyphen#1{%
456   {\escapechar\m@ne
457    \@xa\gdef\@xa\verbhyphen\@xa{\string#1}}}
```

As you see, it's always global.

The breakables

Let's define a \discretionary left brace such that if it breaks, it turns {% at the end of line. We'll use it in almost Knuthian \ttverbatim—it's part of this 'almost'.

```
\breaklbrace 465 \def\breaklbrace{%
466   \discretionary{\type@lbrace\verbhyphen}{}{\type@lbrace}%
467   \yesy}

469 \foone{\catcode`\[=1_\catcode`\{=\active_\catcode`\]=2_\%
470 [%
\dobreaklbrace 471   \def\dobreaklbrace[\catcode`\{=\active
472   \def{%
\breaklbrace 473     [\breaklbrace\gm@lbracehook]]%
474 ]
```

Now we only initialise the hook. Real use of it will be made in gmdoc.

```
478 \relaxen\gm@lbracehook
```

The \bslash macro defined below I use also in more 'normal' T_EXing, e.g., to \typeout some \outer macro's name.

```
483 \foone{\catcode`\!=0_\@makeother\\}%
484 {%
\bslash 485   !def!bslash{\}%
486   }% of \foone.

\breakbslash 489 \def\breakbslash{%
490   \discretionary{\verbhyphen}%
```

491 {\type@bslash}{\type@bslash}\yeshy% it seems that we allow hyphen-
 ation after backslash but hyphenation will be allowed iff \hyphenchar
 % \font is nonnegative.
 494 }% of \breakbslash.

Sometimes line breaking at a backslash may be unwelcome. The basic case, when the first CS in a verbatim breaks at the line end leaving there %, is covered by line 938. For the others let's give the user a counter-crank:

\nobreakbslash 500 \pdef\nobreakbslash{\def\breakbslash{\type@bslash\yeshy}}% to use
 \breakbslash due to the common scoping rules. But for the special case of a backslash open-
 ing a verbatim scope, we deal specially in the line 938.

Analogously, let's provide a possibility of 'nobreaking' the left brace:

\nobreaklbrace 507 \pdef\nobreaklbrace{\def\breaklbrace{\type@lbrace\yeshy}}
 \breaklbrace 510 \foone{\catcode`\!=0\catcode`\=\active}%
 512 {%
 \dobreakbslash 513 !def\dobreakbslash{!catcode`\!=!active\def{\!breakbslash}}%
 \breakbslash 514 }

The macros defined below, \visiblebreakspaces and \xiiclub we'll use in the almost Knuthian macro making verbatim. This 'almost' makes a difference.

\breakablevisspace 523 \def\breakablevisspace{\discretionary{\visiblespace}{}{%
 \visiblespace}}
 The \visiblespace macro is \let in gmutils to \xiispace or to \xxt@visi-
 blespace of xltextra if available.
 527 \foone\obeyspaces% it's just re\catcode'ing.
 528 {%
 \dobreakvisiblespace 529 \newcommand*\dobreakvisiblespace{\def\breakablevisspace}%
 \breakablevisspace \obeyspaces}% \defing it caused a stack overflow disaster with
 gmdoc.
 \dobreakblankspace 531 \newcommand*\dobreakblankspace{\let\space\obeyspaces}%
 532 }
 535 \foone{\@makeother\|}%
 \xiiclub 536 \def\xiiclub{||}}

Almost-Knuthian \ttverbatim

\ttverbatim comes from *The T_EX book* too, but I add into it a L^AT_EX macro changing the \catcodes and make spaces visible and breakable and left braces too.

\ttverbatim 545 \pdef\ttverbatim{%
 546 \let\do=\do@noligs\verbatim@nolig@list
 547 \let\do=\@makeother\dospecials
 548 \dobreaklbrace\dobreakbslash
 549 \dobreakspace
 550 \makeatletter
 554 \ifhmode
 555 \setspaceskip
 556 \fi
 557 \verbatimfont
 558 \xdef\gmv@storedhyphenchar{\the\hyphenchar\font}%

Assignment of the hyphenchar is always global so let the above edefinition be also such.

```
561 \hyphenchar\font=\gmv@hyphenchar
562 \ttverbatim@hook
563 }% of \ttverbatim
```

(2010/08/14, v0.993:) rigid \tt in \ttverbatim changed to redefinable \verbatimfont due to absurd problems with bad fontifying of gmdoc

```
\verbatimfont 569 \def\verbatimfont{\tt}
```

While typesetting stuff in the QX fontencoding I noticed there were no spaces in verbatims. That was because the QX encoding doesn't have any reasonable char at position 32. So we provide a hook in the very core of the verbatim making macros to set proper fontencoding for instance.

```
576 \@emptify\ttverbatim@hook
```

```
\VerbT1 579 \def\VerbT1{\def\ttverbatim@hook{\fontencoding{T1}%
\VerbT      \selectfont}}
\ttverbatim@hook
```

We wish the visible spaces to be the default.

```
583 \let\dobreakspace=\dobreakvisiblespace
```

The core: from shortverb

The below is copied verbatim ;-) from doc.pdf and then is added my slight changes.

```
\MakeShortVerb 592 \def\MakeShortVerb{%
593   \gmu@ifstar
\@shortverbdef 594   {\def\@shortverbdef{\verb*}\@MakeShortVerb}%
\@shortverbdef 595   {\def\@shortverbdef{\verb}\@MakeShortVerb}}
\@MakeShortVerb 598 \def\@MakeShortVerb#1{%
599   \@xa\ifx\csname_cc\string#1\endcsname\relax
600   \@shortverbinfo{Made_}{#1}\@shortverbdef
601   \add@special{#1}%
602   \AddtoPrivateOthers#1% a macro to be really defined in gmdoc.
603   \@xa
604   \xdef\csname_cc\string#1\endcsname{\the\catcode`#1}%
605   \begingroup
606   \catcode`\~\active_\lccode`\~=`#1%
607   \lowercase{%
608     \global\@xa\let
609     \csname_ac\string#1\endcsname~%
610     \@xa\gdef\@xa~\@xa{%
611       \@xa\ifmmode\@xa\string\@xa~%
612       \@xa\else\@xa\afterfi{\@shortverbdef~}\fi}}% This terrible num-
613       ber of \expandafters is to make the shortverb char just other in the
       math mode (my addition).
614   \endgroup
615   \global\catcode`#1\active
616   \else
617   \@shortverbinfo\@empty{#1_ already}{\@empty\verb(*)}%
618   \fi}
\DeleteShortVerb 623 \def\DeleteShortVerb#1{%
```



```

624 \@xa\ifx\csname_cc\string#1\endcsname\relax
625 \@shortvrbinf\@empty{#1_not}{\@empty\verb(*)}%
626 \else
627 \@shortvrbinf{Deleted_}{#1_as}{\@empty\verb(*)}%
628 \rem@special{#1}%
629 \global\catcode`#1\csname_cc\string#1\endcsname
630 \global\@xa\let_\csname_cc\string#1\endcsname_\relax
631 \ifnum\catcode`#1=\active
632 \begingroup
633 \catcode`\~\active_\lccode`\~`#1%
634 \lowercase{%
635   \global\@xa\let_\@xa~%
636   \csname_ac\string#1\endcsname}%
637 \endgroup_\fi_\fi}

```

My little addition

```

\gmvp@packname 641 \@ifpackageloaded{gmdoc}{%
\gmvp@packname 642   \def\gmvp@packname{gmdoc}}{%
\gmvp@packname 643   \def\gmvp@packname{gmverb}}
\@shortvrbinf 646 \def\@shortvrbinf#1#2#3{%
647   \PackageInfo{\gmvp@packname}{%
648     ^^J\@empty_#1\@xa\@gobble\string#2_a_short_reference
649     for_\@xa\string#3}}
\add@special 652 \def\add@special#1{%
653   \rem@special{#1}%
654   \@xa\gdef\@xa\dospecials\@xa
655   {\dospecials_\do_#1}%
656   \@xa\gdef\@xa\@sanitize\@xa
657   {\@sanitize_\@makeoother_#1}}

```

For the commentary on the below macro see the doc package’s documentation. Here let’s only say it’s just amazing: so tricky and wicked use of `\do`. The internal macro `\rem@special` defines `\do` to expand to nothing if the `\do`’s argument is the one to be removed and to unexpandable CSes `\do` and `<do’s argument>` otherwise. With `\do` defined this way the entire list is just globally expanded itself. Analogous hack is done to the `\@sanitize` list.

```

\rem@special 668 \def\rem@special#1{%
669   \def\do##1{%
670     \ifnum`#1=`##1_\else_\@nx\do\@nx##1\fi}%
671   \xdef\dospecials{\dospecials}%
672   \begingroup
673   \def\@makeoother##1{%
674     \ifnum`#1=`##1_\else_\@nx\@makeoother\@nx##1\fi}%
675   \xdef\@sanitize{\@sanitize}%
676   \endgroup}

```

And now the definition of `verbatim` itself. As you’ll see (I hope), the internal macros of it look for the name of the current environment (i.e., `\@currenvir`’s meaning) to set their expectation of the environment’s `\end` properly. This is done to allow the user to define his/her own environments with `\verbatim` inside them. I.e., as with the `verbatim` package, you may write `\verbatim` in the begin definition of your environment and then necessarily `\endverbatim` in its end definition. Of course (or maybe *surprisingly*),

the commands written in the begin definition after `\verbatim` will also be executed at `\begin{environment}`.

```

verbatim 689 \def\verbatim{%
\verbatim 690 \edef\gmv@hyphenpe{\the\hyphenpenalty}%
691 \edef\gmv@exhyphenpe{\the\exhyphenpenalty}%
692 \@beginparpenalty_\predisplaypenalty_\@verbatim
693 \frenchspacing_\gmoveyspaces_\@xverbatim
694 \hyphenpenalty=\gmv@hyphenpe\relax
695 \exhyphenpenalty=\gmv@exhyphenpe
696 \hyphenchar\font=\m@ne

    The line below serves as the delimiter for \verbatim@PitchContents, to discard
    the stuff before it (see l. 774).

702 \gobble\verbatim
703 }% in the LATEX version there's \vobeyspaces instead of \gmoveyspaces.
verbatim* 708 \@namedef{verbatim*}{\@beginparpenalty_\predisplaypenalty_\%
        \@verbatim
709 \@sxverbatim_\% it's the same as \@xverbatim and defines the verbatim end
        (a macro delimited with \end{<curr.envir.>}).
711 }

\endverbatim 713 \def\endverbatim{\@@par
        %% \hyphenchar\font=\gmv@storedhyphenchar % hyphenchar assignments
        are
        %% % always global. And for an entire paragraph works the one last in it so we
        hide it.

717 \ifdim\lastskip_\>\z@
718 \@tempskipa\lastskip_\vskip_\- \lastskip
719 \advance\@tempskipa\parskip_\advance\@tempskipa_\-
        \@outerparskip
720 \vskip\@tempskipa
721 \fi
722 \addvspace\@topsepadd
723 \@endparenv}

        *

726 \n@melet{endverbatim*}{endverbatim}

729 \begingroup_\catcode_\!=0_\%
730 \catcode_\ [=1_\catcode_\]=2_\%
731 \catcode`\{=\active
732 \@makeother\}%
733 \catcode`\=\active%
\@xverbatim 734 !gdef!\@xverbatim[%
735 [!endlinechar!\m@ne_\!everyeof[!\@nx]%
736 !edef!verbatim@currenvir[%
740 !@xa!scantokens!@xa[!\@currenvir]%
742 ]% of \verbatim@currenvir. This macro is defined as the meaning of
        % \@currenvir rescanned. It's done specially for the active star in my
        verbatims. \@currenvir is fully expanded but my active star is \pro|
        tected.
746 !@xa]% and here a little trick with groups:

```

```

747      !@xa!def!@xa!verbatim@currenvir
748      !@xa[!verbatim@currenvir]%
749      !edef!verbatim@edef[%
750      !def!@nx!verbatim@end%
751      ####1!noexpand\end!@nx{%
752      !@xa!unexpanded!@xa[!verbatim@currenvir]%
753      }[%
754      !@nx!verbatim@PitchContents_####1%
755      !@nx!verbatim@PitchContents@Delim% added 2010/8/16
756      ####1!@nx!end[!@currenvir]]}%
757      !verbatim@edef
758      !verbatim@end]%
759      !endgroup
\@sxverbatim 764 \let\@sxverbatim=\@xverbatim
atim@PitchContents@Left 766 \def\verbatim@PitchContents@Left{%
verbatim@PitchContents 767 \long\def\verbatim@PitchContents
768 ##1\@gobble\verbatim_##2\verbatim@PitchContents@Delim
769 }

By default we make \verbatim@PitchContents a gobbler.
772 \verbatim@PitchContents@Left_{}

\VerbatimPitch 774 \pdef\VerbatimPitch{%

But in this declaration scope we make \@xverbatim pitch the contents of verba|
tims in a macro. We use that in gmdoc not to repeat examples' code.

779 \verbatim@PitchContents@Left_{}%
780 \gdef\VerbatimContents{##2}%
781 }%
783 }% of \VerbatimThrow.

F. Mittelbach says the below is copied almost verbatim from LATEX source, modulo
\check@percent.

\@verbatim 788 \def\@verbatim{%

Originally here was just \trivlist_ \item[], but it worked badly in my docu-
ment(s), so let's take just highlights of if.

794 \parsep\parskip

From \@trivlist:

796 \if@noskipsec_ \leavevmode_ \fi
797 \@topsepadd_ \topsep
798 \ifvmode
799 \advance\@topsepadd_ \partopsep
800 \else
801 \unskip_ \par
802 \fi
803 \@topsep_ \@topsepadd
804 \advance\@topsep_ \parskip
805 \@outerparskip_ \parskip

(End of \trivlistlist and \@trivlist highlights.)

807 @@par\addvspace\@topsep

```

```

808 \if@minipage\else\vskip\parskip\fi
810 \advance\@totalleftmargin\verbatimleftskip
811 \parskip\verbatimparskip_ added 2010/6/2
812 \raggedright
813 \leftskip\@totalleftmargin% so many assignments to preserve the list
      thinking for possible future changes. However, we may be sure no internal
      list shall use \@totalleftmargin as far as no inner environments are
      possible in verbatim[*].
819 \@@par% most probably redundant.
820 \@tempswafalse
821 \def\par{% but I don't want the terribly ugly empty lines when a blank line
      is met. Let's make them gmdoc-like i.e., let a vertical space be added as in
      between stanzas of poetry. Originally \if@tempswa\hbox{}\fi, in my
      version will be
826 \ifvmode\if@tempswa\addvspace\stanzaskip\@tempswafalse\fi%
      \fi
827 \@@par
828 \penalty\interlinepenalty_\check@percent}%
829 \everypar{\@tempwattrue\hangindent\verbatimhangindent%
      \hangafter\@ne}% since several chars are breakable, there's
      a possibility of breaking some lines. We wish them to be hanging indented.
832 \obeylines
833 \ttverbatim
834 \verbatim@specials
835 }

```

```

\stanzaskip 837 \@ifundefined{stanzaskip}{\newlength\stanzaskip}{}
838 \stanzaskip=\medskipamount

```

```

\verbatimleftskip 840 \newskip\verbatimleftskip
842 \verbatimleftskip\leftmargini

```

```

\verbatimhangindent 844 \newskip\verbatimhangindent
846 \verbatimhangindent=3em

```

```

\verbatimparskip 848 \newskip\verbatimparskip
849 \verbatimparskip\z@skip

```

```

\check@percent 851 \providecommand*\check@percent{}

```

In the gmdoc package shall it be defined to check if the next line begins with a comment char.

Similarly, the next macro shall in gmdoc be defined to update a list useful to that package. For now let it just gobble its argument.

```

\AddtoPrivateOthers 858 \providecommand*\AddtoPrivateOthers[1]{}

```

Both of the above are \provided to allow the user to load gmverb after gmdoc (which would be redundant since gmdoc loads this package on its own, but anyway should be harmless).

Let's define the 'short' verbatim command.

```

\verb* 867 \def\verb{%
\verb 868 \relax\ifmmode\hbox\else\leavevmode\null\fi
869 \bgroup
870 \ttverbatim
871 \verbatim@specials

```

```

872 \gm@verb@eol
873 \gmu@ifstar
874 {\verb@lasthook\@sverb@chbsl}%
875 {\gmobeyspaces\frenchspacing\verb@lasthook\@sverb@chbsl}}% in
      the LATEX version there's \vobeyspaces instead of \gmobeyspaces.
879 \emptify\verb@lasthook
\@sverb@chbsl 881 \def\@sverb@chbsl#1{\@sverb#1\check@bslash}
\@def@breakbslash 884 \def\@def@breakbslash{\breakbslash}% because \ is \defined as \break|
      bslash not \let.
      For the special case of a backslash opening a (short) verbatim, in which it shouldn't
      be breakable, we define the checking macro.
\DefineTypeChar 890 \DeclareCommand\DefineTypeChar{mmo}{%
% #1 m the char as a CS,
% #2 m short name of the char.
% [#3] o the cs of the char in 'other' catcode.
897 \@namedef{gmu@#2wd@name}{#2wd_of
898 \@xa\meaning\the\font\space_at_\detoken@xa\f@size_pt}
900 \@namedef{gmu@measure#2}{%
901 \unless\ifcsname\csname_gmu@#2wd@name\endcsname\endcsname
902 \gmu@measurewd{#1}% \edefs \gmu@tempa as the width of the char and
% \gmu@tempb as the width of the char among 20 copies of itself.
905 \@xa\gn@melet\csname_gmu@#2wd@name\endcsname{gmu@tempb}% here
      we let the CS with the name contained in \gmu@<char-name>wd@name to
      the expanded value of width of the char measured among copies of it.
914 \fi
915 }% of \gmu@measure<char-name>.
917 \@nameedef{type@#2}{%
918 \@nx\leavevmode
919 \@xanxcs{gmu@measure#2}%
920 \hbox_to_\@nx\csname
921 \@xanxcs{gmu@#2wd@name}\@nx\endcsname
922 {\IfValueTF{#3}{\@nx#3}{\@xanxcs{#2}}}%
923 \@nx\hss}%
924 }% of \type@<char-name>,
925 }% of \DefineTypeChar.
927 \DefineTypeChar\{\bslash}% this defines \type@bslash and its aides \gmu@measurebsl
      and \gmu@bslashwd@name.
930 \DefineTypeChar\{{lbrace}\xiilbrace}% this defines \type@lbrace and
      its auxilia analogous to the above.
\check@bslash 933 \def\check@bslash{%
934 \@ifnextchar\@def@breakbslash
935 {\type@bslash\yeshy\@gobble}% note we allow hyphenation but actually
      this will have effect if \hyphenchar\font allows hyphenation (when it's
      not > 0).
938 {}
942 \let\verb@balance@group\@empty

```

```
\verb@egroup 945 \def\verb@egroup{\global\let\verb@balance@group\@empty
%% \hyphenchar\font=\gmv@storedhyphenchar % \hyphenchar behaves
like \hyphenpenalty etc.: the last one in a paragraph is in charge for entire.
949 \egroup
950 }
```

```
\gm@verb@eol 954 \let\gm@verb@eol\verb@eol@error
```

The latter is a L^AT_EX_{2 ϵ} kernel macro that `\activate`s line end and defines it to close the verb group and to issue an error message. We use a separate CS'cause we are not quite positive to the forbidden line ends idea. (Although the allowed line ends with a forgotten closing shortverb char caused funny disasters at my work a few times.) Another reason is that gmdoc wishes to redefine it for its own queer purpose.

However, let's leave my former 'permissive' definition under the `\verb@eol` name.

```
966 \begingroup
967 \obeylines\obeyspaces%
968 \gdef\verb@eolOK{\obeylines%
\check@percent 969 \def^M{\_check@percent}%
970 }%
971 \endgroup
```

The `\check@percent` macro here is `\provided` to be `\@empty` but in gmdoc employed shall it be.

Let us leave (give?) a user freedom of choice:

```
\verbeolOK 976 \def\verbeolOK{\let\gm@verb@eol\verb@eolOK}
```

And back to the main matter,

```
979 \def\@sverb#1{%
981 \catcode`#1\active\_lccode`~=#1%
982 \gdef\verb@balance@group{\verb@egroup
983 \@latex@error{Illegal\_use\_of\_backslash\_verb\_command}\@ehc}%
984 \aftergroup\verb@balance@group
985 \lowercase{\let~\verb@egroup}% here we make the delimiter to be the
macro closing the verbatim group.
987 }
```

```
\verbatim@nolig@list 989 \def\verbatim@nolig@list{\do\` \do\< \do\> \do\, \do\' \do\~}
```

```
\do@noligs 991 \def\do@noligs#1{%
992 \catcode`#1\active
993 \begingroup
994 \lccode`~=#1\relax
995 \lowercase{\endgroup\def~{\leavevmode\kern\z@\char`#1}}}
```

And finally, what I thought to be so smart and clever, now is just one of many possible uses of a general almost Rainer Schöpf's macro:

```
\dekclubs 1000 \def\dekclubs{\gmu@ifstar{\MakeShortVerb*||}{\MakeShortVerb%
||}}
\olddekclubs 1001 \def\olddekclubs{\OldMakeShortVerb||}
```

But even if a shortverb is unconditional, the spaces in the math mode are not printed. So,

```
\edverbs 1009 \newcommand*\edverbs{%
```

```

1010 \let\gmv@dismath\[ %
1011 \let\gmv@edismath\] %
1012 \def\[{%
1013   \@ifnextac\gmv@disverb\gmv@dismath}%
1014 \relaxen\edverbs}%

\gmv@disverb 1016 \def\gmv@disverb{%
1017   \gmv@dismath
1019   \hbox\bgroup\def\[\{\egroup\gmv@edismath\}}

```

doc- and shortvrb-compatibility

One of minor errors while T_EXing doc.dtx was caused by my understanding of a ‘shortverb’ char: at my settings, in the math mode an active ‘shortverb’ char expands to itself’s ‘other’ version thanks to \string. doc/shortvrb’s concept is different, there a ‘shortverb’ char should work as usual in the math mode. So let it may be as they wish:

```

\old@MakeShortVerb 1031 \def\old@MakeShortVerb#1{%
1032   \@xa@ifx\csname_cc\string#1\endcsname\relax
1033   \@shortvrbinfo{Made_}{#1}\@shortvrbdef
1034   \add@special{#1}%
1035   \AddtoPrivateOthers#1% a macro to be really defined in gmdoc.
1037   \@xa
1038   \xdef\csname_cc\string#1\endcsname{\the\catcode`#1}%
1039   \begingroup
1040   \catcode`\~\active_\lccode`\~`#1%
1041   \lowercase{%
1042     \global\@xa\let\csname_ac\string#1\endcsname~%
1043     \@xa\gdef\@xa~\@xa{%
1044       \@shortvrbdef~}}%
1045   \endgroup
1046   \global\catcode`#1\active
1047   \else
1048   \@shortvrbinfo\@empty{#1_already}{\@empty\verb(*)}%
1049   \fi}

\OldMakeShortVerb 1052 \def\OldMakeShortVerb{\begingroup
1053   \let\@MakeShortVerb=\old@MakeShortVerb
1054   \gmu@ifstar{\eg@MakeShortVerbStar}{\eg@MakeShortVerb}}

\eg@MakeShortVerbStar 1057 \def\eg@MakeShortVerbStar#1{\MakeShortVerb*#1\endgroup}
\eg@MakeShortVerb 1058 \def\eg@MakeShortVerb#1{\MakeShortVerb#1\endgroup}

```

Grey visible spaces

In August 2008 Will Robertson suggested grey spaces for gmdoc. I added a respective option to that package but I like the grey spaces so much that I want provide them for any verbatim environments, so I bring the definition here. The declaration, if put in the preamble, postpones redefinition of \visibleSPACE till \begin{document} to recognise possible redefinition of it when xltextra is loaded.

```

1069 \let\gmd@preambleABD\AtBeginDocument
1070 \AtBeginDocument{\let\gmd@preambleABD\firstofone}

1072 \RequirePackage{xcolor}% for \providecolor

\VisSpacesGrey 1074 \def\VisSpacesGrey{%

```

```

1076 \providecolor{visspacesgrey}{gray}{0.5}%
1077 \gmd@preambleABD{%
1078 \edef\visiblespace{%
1079 \hbox{\@nx\textcolor{visspacesgrey}%
1080 {\@xa\unexpanded\@xa{\visiblespace}}}}%
1081 }}

```

Verbatim specials—CSes in verbatims

```

\verbatim@specials 1085 \pdef\verbatim@specials{% This declaration only defines a bearer of the ‘verba-
    tim specials’.
    % #1 m char for verbatim escape char (for catcode 0), has to be unbraced,
    % #2 m char for verbatim group begin (for catcode 1), has to be unbraced,
    % #3 m char for verbatim group end (for catcode 2), has to be unbraced,
    % [#4] o char for verbatim math shift (for catcode 3),
    % [#5] o char for a shorthand for \metachar.
    % {#6} b (optional braced) additional stuff (commands) to be executed at the
    beginning of the verbatims.
1096 \@bsphack
1097 \begingroup
1098 \let\do\@makeother
1099 \dospecials
1100 \catcode`\_ =10
1101 \verbatim@specials@iii}

\verbatim@specials@iii 1103 \pdef\verbatim@specials@iii#1#2#3{% as you see, we take only first three
    arguments in a despecialized group. It’s to avoid \futurelet of the op-
    tionals’ parser to touch (and thus spoil) subsequent token. Yes, we could
    handle the case of a space or single line end but handling the case of a back-
    slash would be somewhat difficult.
1109 \endgroup
\verbatim@specials@list 1110 \def\verbatim@specials@list{#1#2#3}%
1111 \@ifnextchar{%
1112 {\begingroup\let\do\@makeother\dospecials
1113 \catcode`\_ =9
1114 \verbatim@specials@iv}%
1115 {\addtomacro\verbatim@specials@list{\NoValue\NoValue}%
1116 \verbatim@specials@vi}%
1117 }% of \verbatim@specials@iii.

\verbatim@specials@iv 1119 \pdef\verbatim@specials@iv[#1]{%
1120 \endgroup
1121 \addtomacro\verbatim@specials@list{#1}%
1122 \@ifnextchar{%
1123 {\begingroup\let\do\@makeother\dospecials
1124 \catcode`\_ =9
1125 \verbatim@specials@v}%
1126 {\addtomacro\verbatim@specials@list{\NoValue}%
1127 \verbatim@specials@vi}%
1128 }% of \verbatim@specials@iv.

\verbatim@specials@v 1130 \pdef\verbatim@specials@v[#1]{%
1131 \endgroup
1132 \addtomacro\verbatim@specials@list{#1}%
1133 \verbatim@specials@vi

```



```

1134 }% of \verbatim@specials@v.
\verbatim@specials@vi 1136 \DeclareCommand\verbatim@specials@vi\long{b}
1137 {\addtomacro\verbatim@specials@list{{#1}}}%
1138 \@esphack}

\verbatim@specials 1140 \def\verbatim@specials{% this is the macro that actually sets the chars given
in \verbatim@specials@list as the escape char, group begin and group
end.
1143 \ifdefined\verbatim@specials@list
1144 \@xa\verbatim@specials@\verbatim@specials@list
1145 \fi
1146 }% of \verbatim@specials.

\verbatim@specials@ 1148 \long\def\verbatim@specials@#1#2#3#4#5#6{%
1150 \catcode`#1=0
1151 \protected\@namedef{#1}{#1}%
1152 \catcode`#2=1
1153 \protected\@namedef{#2}{#2}%
1154 \catcode`#3=2
1155 \protected\@namedef{#3}{#3}%
1156 \edef\gmu@tempa{\the\endlinechar}%
1157 \endlinechar\m@ne_ we have to suppress adding of a line end by \scanto|
kens since it would turn into an active char ^^M and raise an error (which
actually did happen).
1160 \scantokens{%
1161 #1let#1bgroup=#2%
1162 #1let#1egroup=#3%
1163 #1catcode#1backquote#1h=6#1relax%
1164 #1pdef#1<h1>#2#1meta#2h1#3#3%
1165 #1catcode#1backquote#1h=11#1relax%
1166 }%
1167 \endlinechar\gmu@tempa\relax
1168 \IfValueT{#4}{%
1169 \catcode`#4=3
1170 \protected\@namedef{#4}{#4}}%
1171 \IfValueT{#5}{%
1172 \begingroup
1173 \lccode`\~=#5\lowercase{\endgroup\let~\metachar}%
1174 \protected\@namedef{#5}{#5}%
1175 \catcode`#5=\active
1176 }% of if value #5.
1177 \PutIfValue{#6}%
1178 }

\noverbatimspecials 1180 \pdef\noverbatimspecials{\let\verbatim@specials@list%
\@undefined}

\GMverbatimspecials 1182 \def\GMverbatimspecials{%
1183 \gmu@ifCSdefined_{\XeTeXversion}%
1185 {\verbatimspecials
1186 <% escape
1187 <<% \bgroup and \egroup
1188 [;]% math shift
1189 [>]% meta-char

```

```

1190     {\def\|\{\metachar{\$vert$}}}%
1191     \makestarlow
1192     \relaxen\'% to provide a CS separator (space is not 10 in verbatims).
1193     \let\>\string
1194     }% of #6.
1195 }%
1196 {}%
1197 }% of \GMverbatimspecials.

```

Partial \verb in arguments

Now command for partial verbatims in arguments of commands:

```

1204 \let\gmu@tempa\all@stars
1205 \@xa\addtomacro\@xa\gmu@tempa\@xa{\all@unders}

1207 \foone{\catcode`\#=\active}
\gmv@hashhalving 1208 {\def\gmv@hashhalving{%
\xiihash         1210     \def#\xihash\@ifnextchar#\gobble{}}%
1211     \catcode`\#=\active}%
1212 }

1214 \foone{\@makeother\^^R}{%
1215     \@xa\DeclareCommand\@xa\scanverb\@xa{%
1217         \@xa_Q\@xa{\gmu@tempa}>Pm}{%
            % #1 Q{*_}
            % #2 m the stuff to be rescanned and typeset verbatim. Note that %
            % will be executed during first scan so at best will disappear.
            Spaces are ignored (because of detokenizers that add a space after a CS) but if you
            declare some \verbatimspecials, then you can use /_ where / de-
            notes the escape char in verbatim.

1226     \begingroup
1227     \gmu@septify
1228     \endlinechar=\m@ne
1229     \@xa\IfIntersect\@xa{\all@stars}{#1}%
1230     {\def\_ {\breakablevispace}}}%
1231     {\let\_ =\space}%
1232     \@xa\IfIntersect\@xa{\all@unders}{#1}%
1233     {}}% We make spaces ignored only if there was no underscore in #1 and if #2
            doesn't contain \_.
1235     \gmu@ifxany\_ {#2}%
1236     {}{\addtomacro\verb@lasthook{\catcode`\_ =9\_}}}%
1237     \addtomacro\verb@lasthook{\gmv@hashhalving\_}%
1238     \@makeother\^^R%
1239     \edef\gmu@tempa{%
1240         \@nx\scantokens{%
1241             \bslash\_verb%
1242             ^^R\detokenize{#2}^^R% we delimit the \verb's argument with 'other'
            % ^^R assuming this char to be used very seldom if at all.

1245         }% of \scantokens,
1246         }\gmu@tempa
1247     \endgroup
1248 }% of \scanverb,
1249 }% of \foone.

```

```

verbDiscretionaryHyphen 1251 \def\verbDiscretionaryHyphen#1#2{%
\gmv@hyphenchar 1254 \def\gmv@hyphenchar{\numexpr#1\relax}%
\gmv@hyphen 1255 \def\gmv@hyphen{#2}%
1256 }

1258 \verbDiscretionaryHyphen{"A6}{|}

1260 \XeTeXifprefix\iftrue

(2010/06/28, v0.94:) due to Will Robertson's remark that recatcoding long (no-
ASCII) dashes works only under XYTeX and LATeX, I embrace them in a XYTeX con-
ditional

\gmu@tempa 1265 \def\gmu@tempa{%
\verbLongDashes 1266 \DeclareCommand\verbLongDashes{
1267 >iT{-}% to memorise which dash we set
1268 B{1.41}\_ % expansion of en-dash
1269 >iT{-}\_ % as above
1270 B{2}\_ % expansion of em-dash
1271 }%
1272 }

\gmu@tempb 1274 \def\gmu@tempb{\catcode`-\active\_catcode`-\active}
1276 \foone{\catcode`-\active\_catcode`-\active}
1277 {%
1278 \edef\gmu@tempa{\@xau\gmu@tempa
1279 {%
1280 \@nx\addtomacro\@nx\ttverbatim@hook{%
1281 \@xau\gmu@tempb
1282 \def\@nx-{\@nx\scalebox{##1}[1]{\string-}}%
1283 \def\@nx-{\@nx\scalebox{##2}[1]{\string-}}%
1284 }%
1285 }%
1286 }%
1287 }%
1288 \gmu@tempa
1290 \fi\_ % of if XYTeX.

```

Note that we have two “hyphens”: one for places where a line may be broken with a comment char and another, provided as `\hyphenchar`, for discretionary hyphens at points where correct T_EX code cannot be broken, such as CS names.

(For my GNU Emacs:) Local Variables: mode: doctex coding: utf-8 End:

```
1336 \endinput
```

End of file ‘gmverb.sty’.

□

Change History

gmverb
General:
Checksum 1040 , 83
gmverb v0.79

\edverbs:
added, 1001
gmverb v0.80
\edverbs:

debugged, i.e. `\hbox` added back and redefinition of `\[, 1001`

`\xiiclub:`
`\ttverbatim@hook` added, 536

gmverb v0.81
General:
`\afterfi` made two-argument (first undelimited, the stuff to be put after `\fi`, and the other, delimited with `\fi`, to be discarded, 1290

gmverb v0.82
General:
Checksum 663 , 83

gmverb v0.83
General:
added a hook in the active left brace definition intended for `gmdoc` automatic detection of definitions (in line 473), 1290
Checksum 666 , 83

gmverb v0.84
General:
Checksum 658 , 83

gmverb v0.85
General:
added restoring of `\hyphenpenalty` and `\exhyphenpenalty` and setting `\hyphenchar=-1`, 1290
Checksum 673 , 83

gmverb v0.87
General:
Checksum 661 , 83
visible space tidied and taken from `xltxtra` if available. `gmutils` required.
The `\xii...` CSes moved to `gmutils`.
The documentation driver moved into the `.sty` file, 1290

gmverb v0.88
General:
Checksum 682 , 83
`\VisSpacesGrey:`
added, or rather moved here from `gmdoc`, 1074

gmverb v0.89
General:
`\dekclubs`, `\dekclubs*` and `\olddekclubs` made more consistent, shorthands for `\MakeShortVerb\|`, `\MakeShortVerb*\|` and `\OldMakeShortVerb\|` respectively., 1290
Checksum 686 , 83

gmverb v0.90
General:
Checksum 684 , 83
some `\(b|e)group` changed to `\(begin|end)group`, 1290

gmverb v0.91
General:
Checksum 686 , 83
put to CTAN on 2008/11/21, 83
`\verbatimleftskip:`
added, 840

gmverb v0.92
General:
Checksum 979 because of `\verbatimspecials`, hyphenation in verbatims, low star in verbatims, kerning of backslash in shrunk fonts, 83
`\breakbslash:`
renamed from `\fixbslash`, 500
`\breaklbrace:`
renamed from `\fixlbrace`, 507
`\ttverbatim:`
added `\makeatletter` to sound with the ‘verbatim specials’, namely to allow control sequences containing `@`, 550

gmverb v0.93
General:
Checksum 1035 because of a bug fix in `\scanverb`(halving the hashes), 83
put to CTAN on 2010/03/04, 83
`\gmv@hashhalving:`
cut out as separate macro, 1208
`\verbDiscretionaryHyphen:`
added to synchronise hyphen chars in `gmdoc`’s documentation, 1251
`\xiihash:`
mandatory argument made long (a bug fix), 1215

gmverb v0.94
General:
Checksum 979 because of wrapping the UTF-8 dashes’ setting in a \XeTeX ’s conditional, 83
put to CTAN on 2010/07/07, 83
`\gmv@hyphen:`
due to Will Robertson’s remark that recatcoding long (no-ASCII) dashes works only under \XeTeX and \LuaTeX , I embrace them in a \XeTeX conditional, 1260

gmverb v0.993
`\ttverbatim:`
rigid `\tt` in `\ttverbatim` changed to redefinable `\verbatimfont` due to absurd problems with bad fontifying of `gmdoc`, 563