Grzegorz Murzynowski

The gmverb Package*

by Grzegorz ‘Natror’ Murzynowski
natror (at) gmail (dot) com
This program is subject to the LaTeX Project Public License.
See http://www.ctan.org/tex-archive/help/Catalogue/licenses.lppl.html for
the details of that license.
LPPL status: "author-maintained".
For documentation please refer to the file(s)
gmverb.{sty,pdf}.

49 \NeedsTeXFormat{LaTeX2e}
50 \ProvidesPackage{gmverb}
51 [2011/10/14 v0.98 After shortvrb (FM) but my way (GM)]
54 \(*\texttt{master}\)
56 (A handful of meta-settings skipped)
83 \(*/\texttt{master}\)

Contents

Intro, usage ........................... 1
The package options ............... 4
Installation ........................ 4
Contents of the gmverb.zip archive 4
Compiling of the documentation ... 4
The code ............................ 5
Preliminaries ...................... 5
The breakables .................... 5
Almost-Knuthian \texttt{verbatim} ... 6
The core: from \texttt{shortvrb} ... 7
\texttt{doc}- and \texttt{shortvrb}-compatibility ... 14
Grey visible spaces ............... 14
Verbatim specials—\texttt{CSes} in \texttt{verbatim} ... 15
Partial \texttt{verb} in arguments ... 17
Change History ..................... 18
Index .............................. 20

Intro, usage

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

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

* This file has version number dated .
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., \{\verb{\verb(\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verb\verbatim

\nogbreakbslash (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: \nobreakbrace.)

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.

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.

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 \TeX\ book’s format, \gmverb provides a macro \dekclubs as a shorthand for \MakeShortVerb\|.

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 shall be restored.

One more difference between \gmverb and shortvrb is that the chars \active by \MakeShortVerb in the math mode behave as if they were ‘other’, so you may type e.g., $2\mid0$ to get 2|0 and + \active this way is in the math mode typeset properly etc.

However, if you don’t like such a conditional behaviour, you may specify some $$’s (any nonempty sequence) mixed with star(s) if you wish as an optional argument to keep the short-verbatim char behave as short-verbatim also in math mode.

For compatibility with \gmdoc and for convenience there is a shorthand for that provided, \OldMakeShortVerb.

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

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)

\[[\verbstanin\verbatim st\verbstanin]]

instead of

\[[\verbstanin\verbatim stanin]]

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 only the visible spaces so affects \verb* and \verbatim* and not the unstarred versions.

\visspacesgrey The colour of the visible spaces is named \visspacesgrey and you can redefine it \xcolor
way.

\verbatim specials

We also provide the \verbatim specials declaration that takes six arguments:
\#1 m a char for verbatim escape char (for catcode 0), has to be unbraced\(^1\),
\#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 \texttt{\meta char}; it has to be in square brackets if present. If provided, e.g., \textgg as I suggest in \verbatim specials, then it itself becomes an active char let-equal to \texttt{\meta}, and a \texttt{CS} made of it, \textgg in this example, becomes \texttt{\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

\verbatim specials \textg{«»} [¿] [ ›] {\def\{|{$\vert$}}

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

|\macro{arg\langle arg. \&n+1\rangle\{[No\}Value\{T/F\}}|

to get

\macro{(arg. \ n + 1)}\{[No\}Value\{T/F\}

Note also that \textgg is a control sequence so it doesn’t delimit the short verbatim ‘\texttt{\}' argument.

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

\nverbatim specials

To turn the ‘verbatim specials’ off write \nverbatim specials, which OCSR too.

Note that although we don’t provide a ‘verbatim superscript’ nor ‘verbatim subscript’, you have the \verb DiscretionaryHyphen declaration that takes two arguments. Broken bar is declared as

\verb DiscretionaryHyphen{\"A6}{\textgg}

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?
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 `\VerbatimContents` 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 TeXnical 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 TeX 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 \LaTeX 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, \TeX will find it), then MakeIndex on the `\jobname.idx` file, and then \LaTeX 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 gmdoccc.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 TeX distribution’s files’ database most probably.

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

\RequirePackage{gmcommand}[2011/10/12]

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 ‘orthodoxy’. 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

\let\verbhyphen\xiiipercent

Provide a declaration for easy changing it. Its argument should be of \langle char \rangle form (a \langle char \rangle_12 is also allowed).

\VerbHyphen

\def\VerbHyphen#1{\VerbHyphen
{\escapechar\m@ne
\@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’.

\breakbrace

\def\breakbrace{%
\discretionary{\type@lbrace\verbhyphen}{}{\type@lbrace}\yeshy%
\foone{\catcode`\[=1 \catcode`\{=\active \catcode`\]=2 }%
[%
\dodobreakbrace{\catcode`\[=\active }
\def{%
[\breakbrace\gm@bracehook\]
%
]
Now we only initialise the hook. Real use of it will be made in gmdoc.

\relaxen\gm@bracehook

The \bslash macro defined below I use also in more ‘normal’ \TeX\-ing, e.g., to \typeout some \outer macro’s name.

\verbhyphen

\def\verbhyphen\xiiipercent

\let!def!bslash{\}%

The basic case, when the first CS in a verbatim breaks at the line end leaving there \%, is covered by line 1006. For the others let’s give the user a counter-crack:
\nobreakslash\breakslash

\nobreakbrace\breakbrace

\nobreakvisiblespace\breakvisiblespace

\nobreakblankspace\breakblankspace

\|\break\|\break

Almost-Knuthian \ttverbatim

\ttverbatim comes from The \TeX book too, but I add into it a \LaTeX macro changing the \catcodes and make spaces visible and breakable and left braces too.

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

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

```
\emptifyttverbatim@hook
\VerbT1\def\VerbT1{\def\ttverbatim@hook{\fontencoding{T1}\selectfont}}
\VerbT\let\VerbT=\ttverbatim@hook
We wish the visible spaces to be the default.
```

The core: from shortvrb

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

```latex
\DeclareCommand\MakeShortVerb{
  \ifmmode\gmV@MakeShortVerbAlsoInMath #2\else\gmV@MakeShortVerbNotInMath #2\fi
}\
\name\glet\gmV@mm\string#1\@firstofone
\edef\gmV@tempA{\strip@bslash{#1}}\@XA{\Name\gdef{gmV@mm\string#1}}\@xa{\@xa\ifmmode\gmV@tempA\else\@firstofone\fi}
```

Depending on whether $ was specified as optional argument to \MakeShortVerb, we define the CS \gmV@mm\langle the char \rangle as either sth. special or sth. usual in the math mode:

```
{\gmV@MakeShortVerbAlsoInMath #2}% extracted to a macro because we reassign it while typesetting \LaTeX Sources.
{\gmV@MakeShortVerbNotInMath #2}% of \MakeShortVerb.
```

```
\pdef\gmV@MakeShortVerbAlsoInMath #1{% If some dollars were specified (paid ;-) ) in #1, then we direct the shortverb char to behave in the math mode the same as in normal text.
  \Name\glet\gmV@mm\string#1\@firstofone
}%
```

```
\pdef\gmV@MakeShortVerbNotInMath #1{% If no dollar was paid ;-) in #1, we get vicious and define shortverb to act in math mode as normal (other) char.
  \edef\gmV@tempA{\strip@bslash (#1)}%
  \@XA{%
    \Name\gdef\gmV@mm\string#1%
  }\@xa{% these two expandafters serve to get “other” version of (unbackslashed) #1.\textbackslash fi
  \ifmmode \gmV@tempA
```

Two subsequent expandafters become part of the definition and serve to gobble or execute the short-verbatim macro.

```
  \@xa \@gobble
  \else
  \@xa \@firstofone
  \fi
}%
```
We store the original catcode of a shortverb char in a CS \texttt{gmV@cc\langle the char\rangle}. Moreover, if the char is originally active (of catcode 13), then we create another CS to store its original meaning, \texttt{gmV@ac\langle the char\rangle}.

\begin{verbatim}
\MakeShortVerb 666 \def\MakeShortVerb#1{% 667   \ifx\csname gmV@cc\string#1\endcsname\relax 668     \AddToPrivateOthers\#1\% a macro to be really defined in gmdoc. 669   \else 670     \def\shortvrbinfo{Made }\verb(#1)\% 671     \add@special{#1}% 672     \AddtoPrivateOthers#1% 673     a macro to be really defined in gmdoc. 674   \fi
\end{verbatim}

My little addition

\begin{verbatim}
\DeleteShortVerb 699 \def\DeleteShortVerb#1{% 700   \ifx\csname gmV@cc\string#1\endcsname\relax 701     \shortvrbinfo{Made }\verb(#1)\% 702   \else 703     \shortvrbinfo{Deleted }\verb(#1)\% 704     \rem@special{#1}% 705     \global\catcode`#1\csname gmV@cc\string#1\endcsname 706     \global\@xa\let\csname gmV@cc\string#1\endcsname\relax 707     \ifnum\catcode`#1=\active 708       \begingroup 709         \catcode`~\active \lccode`~`#1% 710       \lowercase{\global\@xa\let\@xa~\@xa{\csname gmV@ac\string#1\endcsname}} 711       \endgroup 712     \fi 713   \fi
\end{verbatim}

\begin{verbatim}
\ifpackageloaded{gmdoc}{\gmv@packname{gmdoc}}{\gmv@packname{gmverb}}\gmv@packname
\def\shortvrbinfo#1#2#3{% 725 \PackageInfo{\gmv@packname}{#2\#3}\%
\end{verbatim}

8
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.

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{⟨curr.envir.⟩}.

The line below serves as the delimiter for \verbatim@PitchContents, to discard the stuff before it (see l. 850).

The \verbatim* version there’s \@vobeyspaces instead of \gmoveyspaces.

\verbatim* (a macro delimited with \end{(curr.envir,)})
\hyphenchar\font=\gmv@store\hyphenchar

\% always global. And for an entire paragraph works the one last in it so we hide it.

\ifdim\lastskip >\z@
  \@tempskipa\lastskip \vskip -\lastskip
  \advance\@tempskipa\parskip
  \advance\@tempskipa -\@outerparskip
  \vskip\@tempskipa
\fi
\addvspace\@topsepadd
\endparenv
}

\n@melet{endverbatim\textstar}{endverbatim}
\begingroup
\catcode`!\=0
\catcode`/=1
\catcode`\{=2
\@makeother\}
!gdef!@xverbatim[\]
  @xverbatim
![everyeof[!@nx]
  !edef!verbatim@currenvir[!
    @xa!
      scantokens!@xa[![@currenvir]
    ]%
  of \verbatim@currenvir. This macro is defined as the meaning of \@cur\n  renvir rescanned. It's done specially for the active star in my verbatims.
  % \@currenvir is fully expanded but my active star is \textit{protected}.

!@xa]
  and here a little trick with groups:
!@xa!def!@xa!verbatim@currenvir
!@xa[!verbatim@currenvir]
!edef!verbatim@edef[!
  @nx!verbatim@end
###1!noexpand\end!@nx{
  @xa!unexpanded!@xa[![verbatim@currenvir]%
}[
  @nx!verbatim@PitchContents ###1%
  @nx!verbatim@PitchContents@Delim added 2010/8/16
  ###1!@nx!end[![@currenvir]]]
!verbatim@edef
  \verbatim@end
###1
]!
@endgroup
}
\let\@sxverbatim=\@xverbatim
\def\verbatim@PitchContents@Left{
\verbatim@PitchContents@Left
\verbatim@PitchContents
\verbatim@PitchContents
\verbatim@PitchContents
\verbatim@PitchContents

By default we make \verbatim@PitchContents a gobbler.

\verbatim@PitchContents@Left ()
\endverbatim

But in this declaration scope we make \texttt{\@verbatim} pitch the contents of \texttt{verbatim} in a macro. We use that in \texttt{gmdoc} not to repeat examples’ code.

\verbatim@PitchContents@Left {% 
\gdef\VerbatimContents{##2}%
}% of \texttt{\VerbatimThrow}.

F. Mittelbach says the below is copied almost verbatim from \LaTeX\ source, modulo \texttt{\check@percent}.

\verbatim
\def\verbatim{%

Originally here was just \texttt{\trivlist \item[]}, but it worked badly in my document(s), so let’s take just highlights of if.

\verbatim
\verbatim@specials

From \verbatimlist:

\verbatim
\parsep\parskip

\verbatim
\if@noskipsec \leavevmode \fi
\topsepadd \topsep
\ifvmode 
\advance\topsepadd \partopsep
\else
\unskip \par
\fi
\topsep\@topsepadd
\advance\topsep \parskip\@outerparskip \parskip

(End of \verbatimlist and \verbatimlist highlights.)

\verbatim
\@@par\addvspace\topsep
\if@minipage\else\vskip\parskip\fi%
\verbatimimpar% added 2010/6/2
\raggedright
\leftskip\@totalleftmargin%

\verbatim
\obeylines
\ttverbatim
\verbatim@specials

11
In the \texttt{gmdoc} package shall it be defined to check if the next line begins with a comment char. Similarly, the next macro shall in \texttt{gmdoc} be defined to update a list useful to that package. For now let it just gobble its argument.

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

Let’s define the ‘short’ verbatim command.

For the special case of a backslash opening a (short) verbatim, in which it shouldn’t be breakable, we define the checking macro.

\begin{verbatim}
\def\verb{\relax\ifmmode\hbox\else\leavevmode\null\fi
\bgroup\ttverbatim\verbatim@specials\gm@verb@eol\gmu@ifstar
{\verb@lasthook\@sverb@chbsl}\
{\gmobeyspaces\frenchspacing\verb@lasthook\@sverb@chbsl})\% in the \LaTeX\ version there’s \texttt{\@vobeyspaces} instead of \texttt{\gmobeyspaces}.
\emptify\verb@lasthook
\def\@sverb@chbsl#1{\@sverb#1\check@bslash}
\def\@def@breakbslash{reakbslash} because \ is \texttt{defined} as \texttt{\breakbslash} not \texttt{\let}.
\end{verbatim}

For the special case of a backslash opening a (short) verbatim, in which it shouldn’t be breakable, we define the checking macro.
\texttt{\gmu@measurewd(#1)}\% \texttt{edef} \texttt{\gmu@tempa} as the width of the char and \texttt{\gmu@tempb} as the width of the char among 20 copies of itself.

\texttt{\@xa} \texttt{\nameedef{\csname gmu@\#2wd@name\endcsname}{\gmu@tempb} here we let the CS with the name contained in \texttt{\gmu\langle char-name\ranglewd@name} to the expanded value of width of the char measured among copies of it.

\texttt{\@ifnextchar\@def@breakbslash\{}% of \texttt{\gmu@measure\langle char-name\rangle}.

\texttt{\@nameedef{\texttt{type@#2}}{\@nx\leavevmode\@xanxcs{\gmu@measure#2} \hbox\to\@nx\csname \@xanxcs{\gmu@#2wd@name}\@nx\endcsname\@nx#3\@nx\hss} of \texttt{\type@\langle char-name\rangle},

\texttt{\@nameedef{\texttt{type@#2}}{\@nx\leavevmode\@xanxcs{\gmu@measure#2} \hbox\to\@nx\csname \@xanxcs{\gmu@measure\langle\text{char-name}\rangle\langle name\rangle\endcsname} to the expanded value of width of the char measured among copies of it.\}

\texttt{\type@bslash} this defines \texttt{\type@bslash} and its aides \texttt{\gmu@measurebslash} and \texttt{\gmu@bslashwd@name}.

\texttt{\DefineTypeChar\{\bslash\}} this defines \texttt{\type@bslash} and its aides analogous to the above.

\texttt{\DefineTypeChar\{\lbrace\} this defines \texttt{\type@lbrace}} and its auxilia analogous to the above.

\texttt{\check@bslash} \texttt{\let\verb@balance@group\@empty}

The latter is a \LaTeX{} kernel macro that \texttt{\active}ates 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 \texttt{gmdoc} wishes to redefine it for its own queer purpose. However, let’s leave my former ‘permissive’ definition under the \texttt{\verb@eol} name.

\texttt{\let\verb@eol\@empty}

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

\texttt{\begingroup \obeylines \obeyspaces}
And back to the main matter,

\begin{quote}
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:
\end{quote}

\begin{verbatim}
\def\dekclubs\gmu@ifstar{\MakeShortVerb⋆\|}{\MakeShortVerb\|}
\def\olddekclubs{\OldMakeShortVerb\|}
\end{verbatim}

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

\begin{verbatim}
\edef\edverbs{\let\gmv@dismath\[
\let\gmv@edismath\]
\def\[\{
\@ifnextac\gmv@disverb\gmv@dismath
\relax\edef\edverbs}
\edef\gmv@disverb{\gmv@dismath
\hbox\bgroup\def\{\egroup\gmv@edismath}
\end{verbatim}

\begin{quote}
\textbf{doc- and shortvrb-compatibility}
\end{quote}

One of minor errors while \TeXing \texttt{doc.dtx} was caused by my understanding of a ‘shortverb’ char: at my settings, in the math mode an active ‘shortverb’ char expands to its own ‘other’ version thanks to \texttt{\string}. \texttt{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:

\begin{verbatim}
\edef\OldMakeShortVerb{\MakeShortVerb $$}
\end{verbatim}

\begin{quote}
\textbf{Grey visible spaces}
\end{quote}

In August 2008 Will Robertson suggested grey spaces for \texttt{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 \texttt{\visiblespace} till \texttt{\begin{document}} to recognise possible redefinition of it when \texttt{xlttextra} is loaded.
\let\gmd@preambleABD\AtBeginDocument
\AtBeginDocument{\let\gmd@preambleABD\firstofone}
\RequirePackage{xcolor} for \providecolor
\def\VisSpacesGrey{\providecolor{visspacesgrey}{gray}{0.5} \gmd@preambleABD{\edef\visiblespace{\hbox{\@nx\textcolor{visspacesgrey}{\@xa\unexpanded\@xa{\visiblespace}}}}}}

\Verbatim specials—CSes in verbatims
\pdef\verbatim@specials{\% This declaration only defines a bearer of the ‘verbatim 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.
\@bsphack
\begingroup \let\do\@makeother \dospecials \catcode`\=10 \verbatim@specials@iii}
\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 optionals’ 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 backslash would be somewhat difficult.
\endgroup
\verbatim@specials@list
\verbatim@specials@iv
\verbatim@specials@iv}{% of \verbatim@specials@iv.
\verbatim@specials@v\pdef\verbatim@specials@v[#1] {%
\endgroup
\addtomacro\verbatim@specials@list(#1) {
\verbatim@specials@vi
}\verbatim@specials@v}{% of \verbatim@specials@v.
\verbatim@specials@vi\DeclareCommand\verbatim@specials@vi\long{b}
\verbatim@specials@vi{\addtomacro\verbatim@specials@list{{#1}}%
\@esphack}
\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.
\ifdefined\verbatim@specials@list
\@xa\verbatim@specials\verbatim@specials@list
\fi
}\verbatim@specials@#1#2#3#4#5#6{%
\catcode`#1=0
\protected\@namedef{#1}{#1}
\catcode`#2=1
\protected\@namedef{#2}{#2}
\catcode`#3=2
\protected\@namedef{#3}{#3}
\edef\gmu@tempa{\the\endlinechar}
\endlinechar\m@ne% we have to suppress adding of a line end by \scantokens
since it would turn into an active char ^^M and raise an error (which actually
did happen).
\scantokens{%
#1let#1bgroup=#2%
#1let#1egroup=#3%
#1catcode#1backquote#1h=6#1relax%
#1pdef#1<h1>#2#1meta#2h1#3#3%
#1catcode#1backquoteh=11#1relax%
}
\endlinechar\m@ne% \scantokens
\gmuIfValueT{#4}{%
\catcode`#4=3
\protected\@namedef{#4}{#4}}%
\gmuIfValueT{#5}{%
\begingroup
\lccode`~=`#5\lowercase{\endgroup\let~\metachar}%
\protected\@namedef{#5}{#5}
\catcode`#5=\active%
}% of if value #5.
\gmuPutIfValue{#6}%
\noverbatimspecials\pdef\noverbatimspecials\let\verbatim@specials@list\@undefined
\noverbatimspecials\def\GMverbatimspecials{%
\gmu@ifCSdefined{\XeTeXversion}%
\verbatim@specials
% escape
Partial \verb in arguments

Now command for partial verbatims in arguments of commands:

\begin{verbatim}
\let\gmu@tempa\all@stars
\0xa\addtomacro\0xa\gmu@tempa\0xa\{\all@unders\}
\gmv@hashhalfing\xiihash\foone{\catcode`#=\active}
\def\gmv@hashhalfing{\gmv@hashhalfing}
\catcode`#=\active
\foone{\@makeother\^^R}{\@xa\DeclareCommand\@xa\scanverb\@xa{\Q\@xa{%#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 \verbimspecials, then you can use \verb^\ where ^\ denotes the escape char in verbatim.
\begingroup
\gmu@septify
@endlinechar=\m@ne
\0xa\IfIntersect\0xa\{\all@stars\}{#1}{\def\{\breakablevisspace}{}{\let\=\space}
\0xa\IfIntersect\0xa\{\all@unders\}{#1}{\gmu@ifxany\{#2}{}{\addtomacro\verb@lasthook{\catcode`\=9 }}}\addtomacro\verb@lasthook{\gmv@hashhalfing}\\0makeother^^R\edef\gmu@tempa{\@nx\scantokens{\bslash\verb^^R\detokenize{#2}^^R we delimit the \verb's argument with 'other' ^\ assuming this char to be used very seldom if at all.}{\scantokens}}\edef\gmu@tempa{.}
\endgroup
\end{verbatim}

17
\verbDiscretionaryHyphen{\char\hexnumexpr#1\relax}\
\gmv@hyphenchar\gmv@hyphen{#2}

\verbDiscretionaryHyphen{"A6}{}
\ifgmuXeTeX
(2010/06/28, v0.94:) due to Will Robertson’s remark that recatcoding long (no-ASCII) dashes works only under \TeX{} and Lua\TeX{}, I embrace them in a \TeX{} conditional

\DeclareCommand\verbLongDashes{
>iT{–} to memorise which dash we set\nB{1.41} % expansion of em-dash\n>iT{—} as above\nB{2} % expansion of en-dash\n}%
\edef\gmu@tempa{\@xau\gmu@tempa{\@nx\addtomacro\@nx\ttverbatim@hook{\@xau\gmu@tempb\def\@nx\scalebox{##1}[1]{\string–}}\def\@nx\scalebox{##2}[1]{\string—}}}%
\fi of if \TeX{}.

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 \TeX{} code cannot be broken, such as CS names.
\endinput

End of file ‘gmverb.sty’.

Change History

gmverb CheckSum 997 , 83
General: gmverb v0.79
CheckSum 1040 , 83
\verb@edverbs@: debugged, i.e. \verb hbox added back and
redefinition of \verb@[], 1068
\verb@\iiiclub@:
\verb@\ttverbatim@hook added, 564

\verb@gmverb v0.81@ General:
\verb@\afterfi made two-argument (first
undelimited, the stuff to be put after
\verb@\fi, and the other, delimited with
\verb@\fi, to be discarded, 1331

\verb@gmverb v0.82@ General:
CheckSum 663 ,

\verb@gmverb v0.83@ General:
added a hook in the active left brace
definition intended for \verbatim@hook
automatic detection of definitions (in
line 504), 1331
CheckSum 666 ,

\verb@gmverb v0.84@ General:
CheckSum 658 ,

\verb@gmverb v0.85@ General:
added restoring of \verb@hyphenpenalty@ and \verb@exhyphenpenalty@ and setting
\verb@hyphenchar=-1, 1331
CheckSum 673 ,

\verb@gmverb v0.86@ General:
CheckSum 661 ,
visible space tidied and taken from
\verbatim@hook if available. \verbatim@hook required.
The \verbatim@hook Cses moved to \verbatim@hook.
The documentation driver moved into the .sty file, 1331

\verb@gmverb v0.87@ General:
CheckSum 662 ,

\verbatimleftskip@ added,

\verb@gmverb v0.88@ General:
CheckSum 979 because of
\verbatimspecials, hyphenation in
verbatims, low star in verbatims,
kerning of backslash in shrunk fonts, 83
\verbatimleftskip: renamed from \verbatimleftskip, 531
\verbatimleftbrace: renamed from \verbatimleftbrace, 538
\verbatimat added \verbatimat to sound with
the ‘verbatim specials’, namely to
allow control sequences containing \verbatimat, 578

\verb@gmverb v0.89@ General:
CheckSum 1035 because of a bug fix in
\verbatimat(halving the hashes), 83
put to CTAN on 2008/11/21, 83

\verb@gmverb v0.90@ General:
CheckSum 684 ,

\verb@gmverb v0.91@ General:
CheckSum 686 , put to CTAN on 2008/11/21, 83
\verbatimleftskip: added, 913

\verb@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 \verbatimleftskip, 531
\verbatimleftbrace: renamed from \verbatimleftbrace, 538
\verbatimat added \verbatimat to sound with
the ‘verbatim specials’, namely to
allow control sequences containing \verbatimat, 578

\verb@gmverb v0.93@ General:
CheckSum 1035 because of a bug fix in
\verbatimat(halving the hashes), 83
put to CTAN on 2010/03/04, 83
\verbatimleftskip: added, or rather moved here from
\verbatimat, 1249
\verbatimleftbrace: cut out as separate macro, 1249
\verbatimat added to synchronise hyphen chars in
\verbatimat’s documentation, 1292
\verbatimat: mandatory argument made long (a bug
fix), 1256

\verb@gmverb v0.94@ General:
CheckSum 979 because of wrapping
the UTF-8 dashes’ setting in a
XƎTEX’s conditional, 83
put to CTAN on 2010/07/07, 83
\verbatimleftskip: due to Will Robertson’s remark that
recatcoding long (non-ASCII) dashes
works only under XƎTEX and
LuaTeX, I embrace them in a XƎTEX
conditional, 1301

\verb@gmverb v0.95@ General:
put to CTAN on 2011/02/02, 83

\verb@gmverb v0.96@ General:
put to CTAN on 2011/10/14, 83

\verb@gmverb v0.993@ \verbatimat: rigid \verbatimat in \verbatimat changed to
redefinable \verbatimatfont due to
absurd problems with bad fontifying
of \verbatimat, 591
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.