The `genmpage` Package
– Generalized minipages –

Thomas Lotze

2006/09/16

1 Introduction

The purpose of this small package is to make LaTeX's minipages more flexible. It is now possible to define styles which determine many of the design parameters of a minipage. Such styles can be used by calling the `minipage` environment with an additional optional argument.

The issue arose from a usenet discussion in `de.comp.text.tex`: someone needed to typeset the content of all minipages in a document `\raggedright` when he switched to `\raggedright` typesetting of the document's body text. This normally requires either placing a `\raggedright` in every `minipage` environment (which is tedious, error-prone, and has little to do with logical mark-up), or defining a new environment (which is impractical for everyday use because of the optional argument structure of the `minipage` environment).

Other than the justification of a minipage's contents, the minipage styles introduced by this package can be used to preset its font parameters (family, series, shape, size), the horizontal and vertical minipage size, the inner and outer vertical alignment parameters, and the indentation of paragraphs. Furthermore, two options for the vertical alignment are introduced in order to align adjacent minipages with respect to their real (visual) top and bottom margins.

The `genmpage` package has been written for LaTeX 2_ε. As the published experimental code for LaTeX 3 shows, there will be the concept of templates which might well render this package useless. At least, it will be both necessary and convenient to re-implement it in terms of templates.

Please feel free to send suggestions, bug reports, or any comments whatsoever concerning this package and its documentation to the author at `thomas@thomas-lotze.de`, or via usenet news in `de.comp.text.tex` or `comp.text.tex`. The package is developed in a subversion repository at `https://svn.thomas-lotze.de/repos/public/genmpage/`, releases being published on the CTAN.

2 Usage

As already mentioned, the `minipage` environment as redefined by the `genmpage` package takes another optional argument which contains the new parameter settings. In order to preserve the argument structure of the usual `minipage` environment, this new optional argument comes last. This way, if the `genmpage` package is loaded but no values are preset, a `minipage` environment without the new argument acts exactly as if the package was not present in the first place.

2.1 Keys

The new argument is evaluated as a `key=value` list. The usual requirements of the `keyval` package hold. The following keys are defined:

`flush`, `raggedright`, `RaggedRight`, `raggedleft`, `center`: These keys are given without values and determine the justification of the minipages contents in the obvious way. If `RaggedRight` is used, the `ragged2e` package must be loaded. This is not provided for by the `genmpage` package.

`ffamily`, `fseries`, `fshape`: These keys determine the font family, series, and shape of the text inside the minipage. They must be given the same values as the corresponding `NFSS` commands `\fontfamily`, `\fontseries`, and `\fontshape`, resp.

`resetfont`: This key doesn't require a value. If it is given, a `\normalfont` is issued before the other font selection commands.

`fsize`: A key to determine the font size. Possible values are the usual size commands without the backslash, e.g. `fsize=small`.

`width`, `height`, `outer`, `inner`: Width, height, and outer and inner vertical alignment of the minipage (taken as values to these keys). If these keys are set, either directly in the optional argument of the `minipage` environment, in the preamble, or by using styles (see below), they override the values given by the usual `minipage` arguments.

`widtharg`, `heightarg`, `outerarg`, `innerarg`: Not requiring values, these keys stop the respective keys for width, height, and alignment from taking precedence over the usual `minipage` arguments.

`parindent`, `keepparindent`: LaTeX sets `\parindent` to zero within a minipage. If the `parindent` key is set, its value is used for `\parindent` instead. This value can be any LaTeX length. If `keepparindent` is set (without values), the paragraph indentation valid outside the minipage is also used within.

Other than in the new optional argument of the `minipage` environment, all keys can be set by the `keyval` command `\setkeys`, for instance:

```
\setkeys{GenMP}{height=0.3\textheight,resetfont,fshape=it,inner=s}
```

2.2 New options for vertical alignment

Following a suggestion by Donald Arseneau (thanks!), I've introduced two new options for the outer vertical alignment of minipages. With the options `T` and `B`, minipages are aligned with respect to their visual margins as opposed to the baselines of the first or last line of text, resp. This will come in most handy if a minipage starts with an image.

As the `genmpage` package doesn't try to fiddle with LaTeX's way of digesting alignment options, it recognizes the `T` and `B` options, inserts a `\vspace{0pt}` command at the beginning or end of the minipage, and passes on a `t` or `b` option. Therefore, `T` and `B` can only be used as values to the `genmpage` package's `inner` key but not as one of the traditional alignment arguments of minipages.

2.3 Styles

`\defineMPstyle`{⟨*style*⟩}{⟨*definitions*⟩}

A minipage style is more or less a shorthand for a series of `key=value` (or `key`) definitions. Any minipage style defined by `\defineMPstyle` can be used as a key without a value either in the `minipage` argument, `setkey` commands, or even other style definitions. `\defineMPstyle` silently redefines a style already existent. An example:

```
\defineMPstyle{comment}{resetfont,fsize=small,width=0.2\textwidth}
```

There is one style predefined: `\defineMPstyle}{plain}{}`. The `plain` style is called before all other definitions. Redefining it will change the behaviour of all minipages concerning those parameters which are not set either explicitly or by using a style or `setkeys` command. As TEX knows no command for switching back to justified text, things like `\raggedright` cannot be overridden later and should therefore be used in the `plain` style with great care.

3 To do

- Further testing
- Improving the documentation, in particular including a section with usage examples
- Implementing some frame and color features

4 Implementation

```
\newlength\@GenMPparindent

\def\@iiiminipage#1#2[#3]#4{%
  \@ifnextchar[%
    {\@ivminipage{#1}{#2}{#3}{#4}}
    {\@ivminipage{#1}{#2}{#3}{#4}[]}}

\def\@ivminipage#1#2#3#4[#5]{%
  \setkeys{GenMP}{plain,#5}%
  \if@GenMPwidth\else\@GenMPwidth#4\fi
  \if@GenMPheight\else
    \def\@GenMPtempa{#2}\def\@GenMPtempb{\relax}%
    \ifx\@GenMPtempa\@GenMPtempb%
      \let\@GenMPheight=\relax
    \else
      \def\@GenMPheight{#2}%
    \fi
  \fi
  \if@GenMPouter\else\def\@GenMPouter{#1}\fi%
  \if@GenMPinner\else\def\@GenMPinner{#3}\fi%
  \@GenMPparindent\parindent
  \leavevmode
  \@pboxswfalse
  \@tempdima\@GenMPwidth
  \def\@mpargs{{\@GenMPouter}{\@GenMPheight}[\@GenMPinner]{\@GenMPwidth}}%
  \setbox\@tempboxa\vbox\bgroup
    \color@begingroup
```

```
      \hsize\@tempdima
      \textwidth\hsize \columnwidth\hsize
      \@parboxrestore
      \def\@mpfn{mpfootnote}\def\thempfn{\thempfootnote}\c@mpfootnote\z@
      \let\@footnotetext\@mpfootnotetext
      \let\@listdepth\@mplistdepth \@mplistdepth\z@
      \@minipagerestore
      \@setminipage}

\def\@setminipage{%
  \@minipagetrue
  \@GenMPtop
  \@GenMPflush
  \@GenMPresetfont
  \@GenMPffamily\@GenMPfseries\@GenMPfshape\selectfont
  \@GenMPfsize
  \@GenMPsetpi
  \everypar{\@minipagefalse\everypar{}}}

\let\@GenMPendminipage\endminipage
\def\endminipage{%
  \@GenMPbottom
  \@GenMPendminipage}

\let\@GenMPflush=\relax
\define@key{GenMP}{flush}[]{\let\@GenMPflush=\relax}
\define@key{GenMP}{raggedright}[]{\let\@GenMPflush=\raggedright}
\define@key{GenMP}{RaggedRight}[]{\let\@GenMPflush=\RaggedRight}
\define@key{GenMP}{raggedleft}[]{\let\@GenMPflush=\raggedleft}
\define@key{GenMP}{center}[]{\let\@GenMPflush=\centering}

\let\@GenMPffamily=\relax
\let\@GenMPfseries=\relax
\let\@GenMPfshape=\relax
\let\@GenMPresetfont=\relax
\let\@GenMPfsize=\relax
\define@key{GenMP}{ffamily}{\def\@GenMPffamily{\fontfamily{#1}}}
\define@key{GenMP}{fseries}{\def\@GenMPfseries{\fontseries{#1}}}
\define@key{GenMP}{fshape}{\def\@GenMPfshape{\fontshape{#1}}}
\define@key{GenMP}{resetfont}[true]{%
  \def\@GenMPread{#1}\def\@GenMPtempa{true}%
  \ifx\@GenMPread\@GenMPtempa%
    \let\@GenMPresetfont=\normalfont
  \else
    \let\@GenMPresetfont=\relax
  \fi}
\define@key{GenMP}{fsize}{\def\@GenMPfsize{\csname #1\endcsname}}

\let\@GenMPsetpi=\relax
\define@key{GenMP}{keepparindent}[true]{%
  \def\@GenMPread{#1}\def\@GenMPtempa{true}%
  \ifx\@GenMPread\@GenMPtempa%
    \def\@GenMPsetpi{\parindent\@GenMPparindent}%
  \fi}
```

```
\define@key{GenMP}{parindent}{\def\@GenMPsetpi{\parindent#1}}

\newlength\@GenMPwidth
\newif\if@GenMPwidth
\let\if@GenMPwidth\iffalse
\define@key{GenMP}{width}{\let\if@GenMPwidth\iftrue\@GenMPwidth#1}
\define@key{GenMP}{widtharg}[]{\let\if@GenMPwidth\iffalse}
\newif\if@GenMPheight
\let\if@GenMPheight\iffalse
\define@key{GenMP}{height}{\let\if@GenMPheight\iftrue\def\@GenMPheight{#1}}
\define@key{GenMP}{heightarg}[]{\let\if@GenMPheight\iffalse}

\newif\if@GenMPouter
\let\if@GenMPouter\iffalse
\let\@GenMPtop=\relax
\let\@GenMPbottom=\relax
\define@key{GenMP}{outer}{%
  \def\@GenMPread{#1}\def\@GenMPtempa{T}%
  \ifx\@GenMPread\@GenMPtempa%
    \def\@GenMPtop{\vspace{0pt}}%
    \def\@GenMPouter{t}%
  \else
    \def\@GenMPtempa{B}%
    \ifx\@GenMPread\@GenMPtempa%
      \def\@GenMPbottom{\vspace{0pt}}%
      \def\@GenMPouter{b}%
    \else
      \def\@GenMPouter{#1}%
    \fi
  \fi
  \let\if@GenMPouter\iftrue
}
\define@key{GenMP}{outerarg}[]{\let\if@GenMPouter\iffalse}

\newif\if@GenMPinner
\let\if@GenMPinner\iffalse
\define@key{GenMP}{inner}{\let\if@GenMPinner\iftrue\def\@GenMPinner{#1}}
\define@key{GenMP}{innerarg}[]{\let\if@GenMPinner\iffalse}

\def\defineMPstyle#1#2{%
  \define@key{GenMP}{#1}[]{\setkeys{GenMP}{#2}}}

\defineMPstyle{plain}{}
```