LaTeX-package for an easy declaration of functions and variables

Thomas Leineweber

1999/03/25

1 Overview

The package method can be used to easily format method- and variabledeclarations with LaTeX. It is based on work by J. Wahlmann and Robert Garmann.

2 Usage

The package is used as usual:

```
\usepackage[<language>]{method}
```

It defines two new environments: method and data. Method is used to typeset method-declarations, data for variable-declarations. At the moment the two options english and german are defined. With these options it is possible to select the language used to typeset the declarations. In the future some other languages will be added.

3 The environment method

`method` Within the environment method the following commands are defined:

`\head` • `\head{`*Header*`}`: The header of the method.

`\para` • `\para{`*Name*`}{`*Description*`}`: Name and description of a parameter.

`\precond` • `\precond{`*Precondition*`}`: Description of a precondition of the method.

`\descr` • `\descr{`*Description*`}`: Description of the method itself.

`\postcond` • `\postcond{`*postcondition*`}`: Description of a postcondition of the method.

`\error` • `\error{`*Exception*`}`: Error and exceptions.

`\return` • `\return{`*Return value*`}`: Description of the data returned by the method.

`\see` • `\see{`*where*`}`: Cross-References.

These commands have the following in common:

- All parameters are simple texts.
- The sequence of the commands inside the method-environment is not relevant. The parts are typeset automatically.
- Up to 26 \para-commands are allowed inside a method environment. When there are more, a warning will be issued und the following parameters will be ignored.
- The header of the method and the parameters are typeset in a typewriter font.
- If the header is extremely long, it can be typeset in more lines with the following macros (an example is given further down):

```
\headtabbed{<functionname>}     Name of the function
\headpara{<parametername>}      one or more parameter
```

4 The environment data

data The environment data equals to the environment method. The macros \head, \descr and \see can also be used inside a data-environment. Further macros inside a data-environment are:

\init
- \init{*Info*}: Information about the generation of the objekt.

\del
- \del{*Info*}: Information about the release of the object.

5 Examples

In this section some examples for the usage of the environments method and data are shown.

```
\begin{method}
 \head{int div(int a, int b, double \&c);}
 \para{a}{dividend}
 \para{b}{divisor}
 \para{\&c}{result of the division}
 \precond{no preconditions}
 \descr{Divides \texttt{a} by \texttt{b} and gives the result
 in \texttt{c}}
 \postcond{no postconditions}
 \error{no errors}
 \return{\texttt{-1}, when  \texttt{b==0}, else \texttt{0}}
 \see{your favourite mathematics book}
\end{method}

\begin{method}
 \headtabbed{PrimObject()
   \headpara{const Matrix transformation,}
   \headpara{AbstGeometry *geometry = 0,}
   \headpara{MaterialApplication *material = 0,}
   \headpara{AbstBumpMap *bumpMap = 0,}
```

```
   \headpara{Distribution *distribution = 0);}
 \para{transformation}{Transformation matrix}
 \para{*geometry}{\ldots}
 \descr{\ldots}
\end{method}

\begin{data}
 \head{char *name}
 \descr{Name of the user}
\end{data}
\begin{data}
 \head{char *no}
 \descr{Telephone-number of the user}
 \see{Telephone Book}
\end{data}
```

6 Identification und documentation

This package can only be used with LaTeX 2_ε. Therefore make sure, we use no other TeX-format.

```
\NeedsTeXFormat{LaTeX2e}
```

Show the name of the package and its version

```
⟨+method⟩\ProvidesPackage{method}
⟨+method⟩               [1999/03/25 v2.0b
⟨+method⟩                LaTeX-package for method- and
⟨+method⟩                data-descriptions (TL)]
```

We have a specialized class for the documentation.

```
⟨*driver⟩
\documentclass[a4paper]{ltxdoc}
```

Set the specific options for the documentation of the package.

```
\DoNotIndex{\",\\,\addtolength,\begin,\CodelineIndex,\CodelineNumbered}
\DoNotIndex{\def,\DocInput,\documentclass,\DoNotIndex,\EnableCrossrefs}
\DoNotIndex{\end,\fbox,\fboxrule,\hfill,\hspace,\ifcase,\or,\fi}
\DoNotIndex{\ifnum,\fi,\item,\itemindent,\labelsep,\labelwidth}
\DoNotIndex{\leftmargin,\listparindent,\NeedsTeXFormat,\newcommand}
\DoNotIndex{\newcount,\newcounter,\newenvironment,\newlength,\sloppy}
\DoNotIndex{\nopagebreak,\PackageError,\parbox,\parindent,\stepcounter}
\DoNotIndex{\PrintChanges,\PrintIndex,\ProvidesPackage,\RecordChanges}
\DoNotIndex{\setcounter,\setlength,\textbf,\texttt,\usepackage,\vspace}
\DoNotIndex{\settowidth,\textwidth,\topsep}
\CodelineNumbered
\CodelineIndex
\EnableCrossrefs
\RecordChanges
\setcounter{StandardModuleDepth}{1}
\usepackage[T1]{fontenc}
\usepackage[latin1]{inputenc}
```

Give all details.

```
\begin{document}
\DocInput{method.dtx}
```

```
\PrintIndex
\PrintChanges
\end{document}
⟨/driver⟩
```

7 Package internals

At the start of a method-environment the actual textwidth is read and saved for the layout of the description.

The commands for the parts, namely `\head`, `\para`, `\precond`, ..., define internal commands with the names `\meth@head`, `\meth@pa`, `\meth@pb`, ..., `\meth@pz`, `\meth@precond`, ... which are defined with the actual parameters.

At the end of a method-environment all these internal saved data is typeset in a (hopefully) fashionable way.

8 Helping commands

`\meth@paranum` `\meth@headparanum`

The counter `\meth@paranum` counts the number of `\para`-commands within a method-environment:

```
\newcounter{meth@paranum}
```

The counter `\meth@headparanum` stores how many `\headpara`-commands are given within a method-environment:

```
\newcounter{meth@headparanum}
```

`\meth@totwid`

The header will be typeset inside a framed minipage with a width of `\@totwid` (= `\textwidth` - 6mm):

```
\newlength{\meth@totwid}
```

`\meth@indent` `\meth@listdecl`

The descriptions are organized as lists with the following parameters:

```
\def\meth@indent{3.5cm}
\def\meth@listdecl{\labelwidth3cm \labelsep0.5cm
                   \itemindent0cm \leftmargin\meth@indent
                   \topsep0cm \listparindent0cm}
```

`\meth@righttotwid`

The right part of the list has a width of `\meth@righttotwid` (= `\meth@totwid` - `\meth@indent`):

```
\newlength{\meth@righttotwid}
```

`\meth@namewid` `\meth@nameindent` `\meth@rightnamewid`

The following lengths are used for the macro `\headtabbed`:

```
\newlength{\meth@namewid}
\newlength{\meth@nameindent}
\newlength{\meth@rightnamewid}
```

9 Options

`\textsee` `\textinit` `\textdel` `\textreturn` `\textprecond` `\textpostcond` `\textdescr` `\texterror`

Now we have the option-processing. The option defines the language, which will be used to print the textual parts of the descriptions. At the moment only the languages english and german are defined.

First, define the parts for german descriptions

```
\DeclareOption{german}{\def\textsee{Siehe auch:}
                       \def\textinit{Erzeugung:}
                       \def\textdel{Freigabe:}
                       \def\textreturn{R\"uckgabewert:}
                       \def\textprecond{Vorbed.:}
                       \def\textpostcond{Nachbed.:}
                       \def\textdescr{Beschreibung:}
                       \def\texterror{Ausnahmebeh.:}}

```

Now for the english descriptions:

```
\DeclareOption{english}{\def\textsee{see also:}
                        \def\textinit{initialisation:}
                        \def\textdel{disposal:}
                        \def\textreturn{return value:}
                        \def\textprecond{precondition:}
                        \def\textpostcond{postcondition:}
                        \def\textdescr{description:}
                        \def\texterror{exceptions:}}
```

The french descriptions, provided by Jean-Pierre Drucbert:

```
\DeclareOption{french}{\def\textsee{voir aussi:}
                       \def\textinit{initialisation:}
                       \def\textdel{lib\'eration:}
                       \def\textreturn{valeur de retour:}
                       \def\textprecond{pr\'econdition:}
                       \def\textpostcond{postcondition:}
                       \def\textdescr{description:}
                       \def\texterror{exceptions:}}
```

Make the english version the default version and process the options.

```
\ExecuteOptions{english}
\ProcessOptions\relax
```

10 Error-detection

\meth@where
\meth@checkdoubleopen
\meth@checknotopen

The macro \head can be used both in the environments method and data. The following value is used to differentiate if method or data is active. If none is active, the counter is set to 99. Method sets it to 0, data to 1.

```
\newcount\meth@where  \meth@where=99
```

Now define the error messages:

```
\def\meth@checkdoubleopen{
  \ifnum\meth@where<99
     \PackageError{method}%
       {There is an method.sty-environment open!}%
       {}
  \fi
}
\def\meth@checknotopen{
  \ifnum\meth@where=99
     \PackageError{method}%
       {There is no method.sty-environment open!}%
```

```
        {}
  \fi
}
```

11 The environment method

method Now we define the environment method.

```
\newenvironment{method}
{
```

First we check, wether a method or data-environment is open. After that we set `\meth@where` to 0, which shows, that we are inside a method-environment.

```
  \meth@checkdoubleopen
  \meth@where=0
```

Define the lengths used for the typesetting of the method.

```
  \setlength{\meth@totwid}{\textwidth}
  \addtolength{\meth@totwid}{-6mm}
  \setlength{\meth@righttotwid}{\meth@totwid}
  \addtolength{\meth@righttotwid}{-\meth@indent}
```

The right column is not very wide. Therfore use `\sloppy`.

```
  \sloppy
```

All parts are defined to nothing.

```
  \def\meth@head{}
  \def\meth@headtabbed{}
  \setcounter{meth@headparanum}{0}
   \def\meth@hpa{}\def\meth@hpb{}\def\meth@hpc{}\def\meth@hpd{}
   \def\meth@hpe{}\def\meth@hpf{}\def\meth@hpg{}\def\meth@hph{}
   \def\meth@hpi{}\def\meth@hpj{}\def\meth@hpk{}\def\meth@hpl{}
   \def\meth@hpm{}\def\meth@hpn{}\def\meth@hpo{}\def\meth@hpp{}
   \def\meth@hpq{}\def\meth@hpr{}\def\meth@hps{}\def\meth@hpt{}
   \def\meth@hpu{}\def\meth@hpv{}\def\meth@hpw{}\def\meth@hpx{}
   \def\meth@hpy{}\def\meth@hpz{}
  \setcounter{meth@paranum}{0}
   \def\meth@pa{}\def\meth@pb{}\def\meth@pc{}\def\meth@pd{}
   \def\meth@pe{}\def\meth@pf{}\def\meth@pg{}\def\meth@ph{}
   \def\meth@pi{}\def\meth@pj{}\def\meth@pk{}\def\meth@pl{}
   \def\meth@pm{}\def\meth@pn{}\def\meth@po{}\def\meth@pp{}
   \def\meth@pq{}\def\meth@pr{}\def\meth@ps{}\def\meth@pt{}
   \def\meth@pu{}\def\meth@pv{}\def\meth@pw{}\def\meth@px{}
   \def\meth@py{}\def\meth@pz{}
  \def\meth@precond{}
  \def\meth@descr{}
  \def\meth@postcond{}
  \def\meth@error{}
  \def\meth@return{}
  \def\meth@see{}
```

Now for the end of the environment. The first line in a paragraph is not indented and a small space is made above the header.

```
}{
  \parindent0cm
  \vspace{2mm}
```

now a sorted list of all given parts inside the environment. The user has to see, that only one of `\meth@head` and `\meth@headtabbed` is used.

```
\meth@head\meth@headtabbed
\nopagebreak[4]
\meth@pa \meth@pb \meth@pc \meth@pd \meth@pe \meth@pf \meth@pg
\meth@ph \meth@pi \meth@pj \meth@pk \meth@pl \meth@pm \meth@pn
\meth@po \meth@pp \meth@pq \meth@pr \meth@ps \meth@pt \meth@pu
\meth@pv \meth@pw \meth@px \meth@py \meth@pz
\meth@precond
\meth@descr
\meth@postcond
\meth@error
\meth@return
\meth@see
```

Now set `\meth@where` back to 99.

```
\meth@where=99
```

This is the end of the definition of the environment method.

```
}
```

12 The environment data

`data` The environment data is nearly equivalent to the environment method.

```
\newenvironment{data}
{
  \meth@checkdoubleopen
  \meth@where=1
  \setlength{\meth@totwid}{\textwidth}
  \addtolength{\meth@totwid}{-6mm}
  \sloppy
  \def\meth@head{}
  \def\meth@descr{}
  \def\meth@init{}
  \def\meth@del{}
  \def\meth@see{}
}{
  \parindent0cm
  \vspace{2mm}
  \meth@head
  \nopagebreak
  \meth@descr
  \meth@init
  \meth@del
  \meth@see
  \meth@where=99
}
```

13 Macros for the parts inside the environments

The definitions for the parts of the environments data and method

13.1 \head

\head The macro \head is used to typeset the header of the method or the definition of the variable.

```
\newcommand{\head}[1]{
```

First check, if the environment is active at the moment.

```
  \meth@checknotopen
```

If \meth@where is set to 0, the environment method is active.

```
  \ifnum\meth@where=0
    \def\meth@head{
```

The code to typeset the header.

```
      {\setlength{\fboxrule}{0.2mm}%
       \fbox{\parbox{\meth@indent}{\hfill}
             \begin{minipage}{\meth@righttotwid}
               {\parindent-\meth@indent \texttt{#1}}
             \end{minipage}
            }}
    }
  \fi%
```

If \meth@where is set to 1, the environment data is active.

```
  \ifnum\meth@where=1
    \def\meth@head{
```

The code to typeset the header.

```
      {\setlength{\fboxrule}{0.1mm}%
       \fbox{\hspace{2mm}\begin{minipage}{\meth@totwid}
         \texttt{#1}
       \end{minipage}}
      }
    }
  \fi%
}
```

13.2 \headtabbed

\headtabbed \headtabbed takes care of the first line, which will be formatted in the header of the method and defines the macro \meth@headtabbed, which does the output.

```
\newcommand{\headtabbed}[1]{
  \meth@checknotopen
  \def\meth@headtabbed{
     \setlength{\fboxrule}{0.2mm}%
     \settowidth{\meth@namewid}{\texttt{#1}}%
     \setlength{\meth@rightnamewid}{\meth@totwid}
     \addtolength{\meth@rightnamewid}{-\meth@namewid}
     \setlength{\meth@nameindent}{\meth@namewid}
     \addtolength{\meth@nameindent}{2mm}
     \fbox{\parbox{\meth@nameindent}{\hfill}%
             \begin{minipage}{\meth@rightnamewid}
                \parindent-\meth@namewid
                \texttt{#1\meth@hpa
                  \meth@hpb
```

```
                    \meth@hpc
                    \meth@hpd
                    \meth@hpe
                    \meth@hpf
                    \meth@hpg
                    \meth@hph
                    \meth@hpi
                    \meth@hpj
                    \meth@hpk
                    \meth@hpl
                    \meth@hpm
                    \meth@hpn
                    \meth@hpo
                    \meth@hpp
                    \meth@hpq
                    \meth@hpr
                    \meth@hps
                    \meth@hpt
                    \meth@hpu
                    \meth@hpv
                    \meth@hpw
                    \meth@hpx
                    \meth@hpy
                    \meth@hpz}%
                \end{minipage}%
            }
  }
}
```

13.3 \headpara

\meth@defheadpara

\meth@defheadpara searches for the first empty macro of \meth@hpa, ..., \meth@hpz. In this macro the new line can be saved. This macro is used by \headpara.

```
\newcommand{\meth@defheadpara}[1]{
  \ifcase\value{meth@headparanum}
    \def\meth@hpa{#1}  \or
    \def\meth@hpb{\\#1}  \or
    \def\meth@hpc{\\#1}  \or
    \def\meth@hpd{\\#1}  \or
    \def\meth@hpe{\\#1}  \or
    \def\meth@hpf{\\#1}  \or
    \def\meth@hpg{\\#1}  \or
    \def\meth@hph{\\#1}  \or
    \def\meth@hpi{\\#1}  \or
    \def\meth@hpj{\\#1}  \or
    \def\meth@hpk{\\#1}  \or
    \def\meth@hpl{\\#1}  \or
    \def\meth@hpm{\\#1}  \or
    \def\meth@hpn{\\#1}  \or
    \def\meth@hpo{\\#1}  \or
    \def\meth@hpp{\\#1}  \or
    \def\meth@hpq{\\#1}  \or
```

```
    \def\meth@hpr{\\#1}  \or
    \def\meth@hps{\\#1}  \or
    \def\meth@hpt{\\#1}  \or
    \def\meth@hpu{\\#1}  \or
    \def\meth@hpv{\\#1}  \or
    \def\meth@hpw{\\#1}  \or
    \def\meth@hpx{\\#1}  \or
    \def\meth@hpy{\\#1}  \or
    \def\meth@hpz{\\#1}  \or
    \PackageError{method}%
       {Too many parameters in method-environment !}{}
  \fi
  \stepcounter{meth@headparanum}
}
```

`\meth@defheadpara` Now for the definition of the macro `\headpara`. Check first, if the correspondent environment is open. Then use the macro `\meth@defheadpara`.

```
\newcommand{\headpara}[1]{
  \meth@checknotopen
  \meth@defheadpara{#1}
}
```

13.4 \para

`\meth@defpara`

```
\newcommand{\meth@defpara}[1]{
  \ifcase\value{meth@paranum}
    \def\meth@pa{#1}  \or
    \def\meth@pb{#1}  \or
    \def\meth@pc{#1}  \or
    \def\meth@pd{#1}  \or
    \def\meth@pe{#1}  \or
    \def\meth@pf{#1}  \or
    \def\meth@pg{#1}  \or
    \def\meth@ph{#1}  \or
    \def\meth@pi{#1}  \or
    \def\meth@pj{#1}  \or
    \def\meth@pk{#1}  \or
    \def\meth@pl{#1}  \or
    \def\meth@pm{#1}  \or
    \def\meth@pn{#1}  \or
    \def\meth@po{#1}  \or
    \def\meth@pp{#1}  \or
    \def\meth@pq{#1}  \or
    \def\meth@pr{#1}  \or
    \def\meth@ps{#1}  \or
    \def\meth@pt{#1}  \or
    \def\meth@pu{#1}  \or
    \def\meth@pv{#1}  \or
    \def\meth@pw{#1}  \or
    \def\meth@px{#1}  \or
    \def\meth@py{#1}  \or
    \def\meth@pz{#1}  \or
```

```
    \PackageError{method}%
    {Too many parameters in method.sty-environment !}
  \fi
  \stepcounter{meth@paranum}
}
```

\para Here the definition for \para:

```
\newcommand{\para}[2]{
  \meth@checknotopen
  \meth@defpara{\begin{list}{\texttt{#1}}{\meth@listdecl}
    \item #2
    \end{list}}
}
```

13.5 The other macros

\precond The other macros are very simple. The create a list-environment and put their data inside of this list.

```
\newcommand{\precond}[1]{
  \meth@checknotopen
  \def\meth@precond{\begin{list}{\textbf{\textprecond}}{\meth@listdecl}
    \item #1
    \end{list}}
}
```

\postcond

```
\newcommand{\postcond}[1]{
  \meth@checknotopen
  \def\meth@postcond{\begin{list}{\textbf{\textpostcond}}{\meth@listdecl}
      \item #1
    \end{list}}
}
```

\descr

```
\newcommand{\descr}[1]{
  \meth@checknotopen
  \def\meth@descr{\begin{list}{\textbf{\textdescr}}{\meth@listdecl}
      \item #1
    \end{list}}
}
```

\error

```
\newcommand{\error}[1]{
  \meth@checknotopen
  \def\meth@error{\begin{list}{\textbf{\texterror}}{\meth@listdecl}
      \item #1
    \end{list}}
}
```

\return

```
\newcommand{\return}[1]{
  \meth@checknotopen
  \def\meth@return{\begin{list}{\textbf{\textreturn}}{\meth@listdecl}
```

```
    \item #1
    \end{list}}
}
```

\see

```
\newcommand{\see}[1]{
  \meth@checknotopen
  \def\meth@see{\begin{list}{\textbf{\textsee}}{\meth@listdecl}
    \item #1
    \end{list}}
}
```

\init

```
\newcommand{\init}[1]{
  \meth@checknotopen
  \def\meth@init{\begin{list}{\textbf{\textinit}}{\meth@listdecl}
    \item #1
    \end{list}}
}
```

\del

```
\newcommand{\del}[1]{
  \meth@checknotopen
  \def\meth@del{\begin{list}{\textbf{\textdel}}{\meth@listdecl}
    \item #1
    \end{list}}
}
```

Index

Numbers written in italic refer to the page 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.

Change History

v1.0

General: First version by J. Wahlmann 1

v1.1

General: Facelifting by R. Garmann 1

init and del for the environment data added 1

New environment data for variables 1

Test for the use of the special commands outside the environments 1

method: Abzug von \meth@totwid von 4 mm auf 6 mm erhhöht .. 6

v1.2

General: Change documentation to a dtx-file 1

v1.3

General: Documentation changes . 1

data: Change \meth@totwid from 4 mm to 6 mm. 7

\del: \bf replaced by \textbf. . 12

\descr: \bf replaced by \textbf. 11

\error: \bf replaced by \textbf. 11

method: Minimal change in the format of typesetting 7

\head: \tt replaced by \texttt. . 8

\headtabbed: \tt replaced by \texttt. 8