The picture package

Heiko Oberdiek
<oberdiek@uni-freiburg.de>

2006/08/26 v1.0

Abstract

There are macro and environment arguments that expect numbers that will internally be multiplicated with `\unitlength`. This package extends the syntax of these arguments that dimens with calculation support can be added for these arguments.

Contents

1 User interface

1.1 Introduction

The environment `picture` and macros such as `\put`, `\line`, `\vector` and other macros have arguments that expect numbers that are used as factor for `\unitlength`. This package redefines such macros and adds code that detects whether such an argument is given as number or as length. In the latter case, the length is used directly without multiplying with `\unitlength`.

1.2 Options

Depending on the available features, also length expressions can be given. Option calc loads package calc. Then expressions of these package may be used. Otherwise etex wraps the length argument inside `\dimexpr`...`\relax`, if ε-TEX is available. Otherwise option plain uses plain assignments without calculation support.

The default is calc if package calc is loaded before package picture. If you specify option calc the loading of calc is ensured. Otherwise package picture looks whether `\dimexpr` is available and uses then option etex as default. If ε-TEX also could not be found, then plain is used.

1.3 Example

```
⟨*example⟩
\documentclass{article}

\usepackage[calc]{picture}

\begin{document}

\setlength{\unitlength}{1pt}

\begin{picture}(\widthof{Hello World}, 10mm)
  \put(0, 0){\makebox(0,0)[lb]{Hello World}}
  \put(0, \heightof{Hello World} + \fboxsep){%
    \line(1, 0){\widthof{Hello World}}%
  }
  \put(\widthof{Hello World}, 10mm){%
    \line(0, -1){10mm}%
  }
\end{picture}%

\end{document}
⟨/example⟩
```

1.4 Supported packages

Package pspicture is supported, but it must be loaded before package picture.

New macros can be supported by `\picture@redefine`. The first argument is the macro which contains the arguments in its parameter text that you want to support by package picture. The second argument contains the parameter text. Change # to & for the arguments in question. Examples (already used by package picture):

```
\picture@redefine\put{(&1,&2)}
\picture@redefine\line{(#1,#2)&3}
```

2 Implementation

2.1 Identification

```
⟨*package⟩
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{picture}%
  [2006/08/26 v1.0 Dimens for picture macros (HO)]
```

2.2 Options

```
\def\Pc@calcname{calc}
\def\Pc@etexname{etex}
\def\Pc@plainname{plain}
```

\Pc@method Macro \Pc@method stores the method to use for calculations. Check which features are available and set the default for \Pc@method.

```
\@ifpackageloaded{calc}{%
  \let\Pc@method\Pc@calcname
}{%
  \begingroup\expandafter\expandafter\expandafter\endgroup
  \expandafter\ifx\csname dimexpr\endcsname\relax
    \let\Pc@method\Pc@plainname
  \else
    \let\Pc@method\Pc@etexname
  \fi
}

\DeclareOption{plain}{%
  \let\Pc@method\Pc@plainname
}
\DeclareOption{etex}{%
  \begingroup\expandafter\expandafter\expandafter\endgroup
  \expandafter\ifx\csname dimexpr\endcsname\relax
    \PackageError{picture}{%
      e-TeX is not available%
    }\@ehc
  \else
    \let\Pc@method\Pc@etexname
  \fi
}
\DeclareOption{calc}{%
  \let\Pc@method\Pc@calcname
}
\ProcessOptions*
\begingroup
  \let\on@line\@empty
  \PackageInfo{picture}{Calculation method: \Pc@method}
\endgroup
```

2.3 Calculation method

```
\ifx\Pc@method\Pc@calcname
  \RequirePackage{calc}%
\fi
```

2.3.1 Method calc

```
\ifx\Pc@method\Pc@calcname
  \def\Pc@tokslength#1{%
    \begingroup
      \let\calc@error\Pc@calc@error
      \setlength\dimen@{#1\unitlength}\Pc@next\Pc@nil{#1}%
  }%
  \let\PcOrg@calc@error\calc@error
  \def\Pc@calc@error#1{%
    \expandafter\ifx\expandafter\unitlength\noexpand#1\relax
      \def\calc@next##1!{%
        \endgroup
        \aftergroup\afterassignment
        \aftergroup\Pc@next
      }%
    \else
      \@ReturnAfterFi{%
        \PcOrg@calc@error{#1}%
      }%
    \fi
  }%
```

```
  \expandafter\@firstofone
\else
  \expandafter\@gobble
\fi
{%
  \long\def\@ReturnAfterFi#1\fi{\fi#1}%
}
```

2.3.2 Method etex

```
\ifx\Pc@method\Pc@etexname
  \def\Pc@tokslength#1{%
    \begingroup
      \afterassignment\Pc@next
      \dimen@=\dimexpr#1\unitlength\Pc@nil{#1}%
  }%
\fi
```

2.3.3 Method plain

```
\ifx\Pc@method\Pc@plainname
  \def\Pc@tokslength#1{%
    \begingroup
      \afterassignment\Pc@next
      \dimen@=#1\unitlength\Pc@nil{#1}%
  }%
\fi
```

2.3.4 Help macros

```
\def\Pc@next#1\Pc@nil#2{%
  \ifx\\#1\\%
    \endgroup
    \Pc@addtoks{{#2}}%
  \else
    \expandafter\endgroup
    \expandafter\Pc@addtoks\expandafter{%
      \expandafter{\the\dimen@\@gobble}%
    }%
  \fi
}
```

`\Pc@nil` `\Pc@nil` must not have the meaning of `\relax` because of `\dimexpr`.

```
\let\Pc@nil\message
```

`\Pc@addtoks`

```
\def\Pc@addtoks#1{%
  \toks@=\expandafter{\the\toks@#1}%
}
```

`\Pc@init`

```
\def\Pc@init#1{%
  \begingroup
    \toks@={#1}%
}
```

`\Pc@finish`

```
\def\Pc@finish#1{%
  \expandafter\endgroup
  \expandafter#1\the\toks@
}
```

2.4 Redefinitions

\picture@redefine #1: command name
#2: parameter text, length parameter with & instead of #

```
\def\picture@redefine#1#2{%
  \begingroup
    \edef\reserved@a{%
      \noexpand\noexpand
      \expandafter\noexpand
          \csname PcOrg@\expandafter\@gobble\string#1\endcsname
    }%
    \toks0{#1}%
    \Pc@first#2&0%
}
```

\Pc@first

```
\def\Pc@first#1&{%
  \toks1={#1}%
  \toks2={\Pc@init{#1}}%
  \Pc@scanlength
}
```

\Pc@scanlength #1: number of length parameter or zero

```
\def\Pc@scanlength#1{%
  \ifcase#1 %
    \expandafter\Pc@last
  \else
    \toks1=\expandafter{\the\toks1 ###1}%
    \toks2=\expandafter{\the\toks2 \Pc@tokslength{###1}}%
    \expandafter\Pc@scannext
  \fi
}
```

\Pc@scannext

```
\def\Pc@scannext#1&{%
  \ifx\\#1\\%
  \else
    \toks1=\expandafter{\the\toks1 #1}%
    \toks2=\expandafter{\the\toks2 \Pc@addtoks{#1}}%
  \fi
  \Pc@scanlength
}
```

\Pc@last

```
\def\Pc@last{%
  \edef\x{%
    \endgroup
    \let\reserved@a\the\toks0 %
    \def\the\toks0 \the\toks1 {%
      \the\toks2 %
      \noexpand\Pc@finish\reserved@a
    }%
  }%
  \x
}
```

2.4.1 LaTeX base macros

```
\picture@redefine\@picture{(&1,&2)(&3,&4)}
\picture@redefine\put{(&1,&2)}
\picture@redefine\multiput{(&1,&2)}
\picture@redefine\@multiput{(&1,&2)}
```

```
\picture@redefine\line{(#1,#2)&3}
\picture@redefine\vector{(#1,#2)&3}
\picture@redefine\dashbox{&1(&2,&3)}
\picture@redefine\@circle{&1}
\picture@redefine\@dot{&1}
\picture@redefine\@bezier{#1(&2,&3)(&4,&5)(&6,&7)}
\picture@redefine\@imakepicbox{(&1,&2)}
```

2.4.2 Package pspicture

Package pspicture changes the signature of `\@oval` by adding an optional argument.

```
\@ifpackageloaded{pspicture}{%
  \picture@redefine\@oval{[&1](&2,&3)}
  \picture@redefine\Line{(&1,&2)}
  \picture@redefine\Curve{(&1,&2)}
  \picture@redefine\Vector{(&1,&2)}
}{
  \picture@redefine\@oval{(&1,&2)}
}
```

⟨/package⟩

3 Installation

CTAN. This package is available on CTAN[1]:

CTAN:macros/latex/contrib/oberdiek/picture.dtx The source file.

CTAN:macros/latex/contrib/oberdiek/picture.pdf Documentation.

Unpacking. The `.dtx` file is a self-extracting docstrip archive. The files are extracted by running the `.dtx` through plain-TEX:

```
tex picture.dtx
```

TDS. Now the different files must be moved into the different directories in your installation TDS tree (also known as `texmf` tree):

```
picture.sty          →  tex/latex/oberdiek/picture.sty
picture.pdf          →  doc/latex/oberdiek/picture.pdf
picture-example.tex  →  doc/latex/oberdiek/picture-example.tex
picture.dtx          →  source/latex/oberdiek/picture.dtx
```

If you have a `docstrip.cfg` that configures and enables docstrip's TDS installing feature, then some files can already be in the right place, see the documentation of docstrip.

Refresh file databases. If your TEX distribution (teTEX, mikTEX, ...) rely on file databases, you must refresh these. For example, teTEX users run `texhash` or `mktexlsr`.

3.1 Some details for the interested

Attached source. The PDF documentation on CTAN also includes the `.dtx` source file. It can be extracted by AcrobatReader 6 or higher. Another option is pdftk, e.g. unpack the file into the current directory:

```
pdftk picture.pdf unpack_files output .
```

[1] ftp://ftp.ctan.org/tex-archive/

Unpacking with LaTeX. The `.dtx` chooses its action depending on the format:

plain-TeX: Run docstrip and extract the files.

LaTeX: Generate the documentation.

If you insist on using LaTeX for docstrip (really, docstrip does not need LaTeX), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{picture.dtx}
```

Do not forget to quote the argument according to the demands of your shell.

Generating the documentation. You can use both the `.dtx` or the `.drv` to generate the documentation. The process can be configured by the configuration file `ltxdoc.cfg`. For instance, put this line into this file, if you want to have A4 as paper format:

```
\PassOptionsToClass{a4paper}{article}
```

An example follows how to generate the documentation with pdfLaTeX:

```
pdflatex picture.dtx
makeindex -s gind.ist picture.idx
pdflatex picture.dtx
makeindex -s gind.ist picture.idx
pdflatex picture.dtx
```

4 History

[2006/08/26 v1.0]

- First released version. (First start of the project was June/July 2002.)

5 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.