

```

% Copyright 2006 by Till Tantau
%
% This file may be distributed and/or modified
%
% 1. under the LaTeX Project Public License and/or
% 2. under the GNU Public License.
%
% See the file doc/generic/pgf/licenses/LICENSE for more details.

\ProvidesFileRCS $Header: /cvsroot/pgf/pgf/generic/pgf/modules/pgfmoduleplot.code.tex,v 1.8
2010/10/22 17:34:17 ludewich Exp $

% PGF's plotting interface works as follows:
%
% In order to plot something, two things need to be done. First, you
% need to provide the coordinates (obviously) of the points that
% should be plotted. The coordinates are given via a long stream of
% commands. These commands are \pgfplotstreamstart, which is
% given exactly once at the beginning, \pgfplotstreampoint, of which there
% are numerous in the middle, \pgfplotstreamspecial, of which there may be
% numerous in the middle, and \pgfplotstreamend, which must be given
% at the end. Between these commands arbitrary other commands may be
% given. Here is an example:
%
% ...
% \pgfplotstreamstart
% \pgfplotstreampoint{\pgfpointxy{0}{0}}
% \pgfplotstreampoint{\pgfpointxy{1}{1}}
% \pgfplotstreampoint{\pgfpointxy{2}{4}}
% \relax
% \pgfplotstreampoint{\pgfpointxy{3}{9}}
% \pgfplotstreamspecial{some handler-dependent special stuff}
% \pgfplotstreamend
%
% By themselves, the \pgfplotstreamxxx commands do not do anything by
% default. Rather, to "use" such a stream, you must first install a
% stream handler. For example, the "lineto" handler will simply
% translate every \pgfplotstreampoint into a \pgfpathlineto.
%
% Example:
%
% \pgfpathmoveto{\pgfpointorigin}
%
% \pgfplotohandlerlineto

```

```
% \pgfplotstreamstart
% \pgfplotstreampoint{\pgfpointxy{0}{0}}
% \pgfplotstreampoint{\pgfpointxy{1}{1}}
% \pgfplotstreampoint{\pgfpointxy{2}{4}}
% \relax
% \pgfplotstreampoint{\pgfpointxy{3}{9}}
% \pgfplotstreamend
```

```
% The stream commands actually call their ``internal" versions, which
% are set by the handlers:
```

```
\def\pgfplotstreamstart{\pgf@plotstreamstart}
\def\pgfplotstreampoint#1{\gdef\pgfplotlastpoint{#1}\pgf@plotstreampoint{#1}}
\def\pgfplotstreamspecial{\pgf@plotstreamspecial}
\def\pgfplotstreamend{\pgf@plotstreamend}
```

```
% Sets the action taken for the first point of a plot to a lineto.
```

```
%
```

```
% Description:
```

```
%
```

```
% For certain handlers it makes sense either the start a plot by
% moving to the first point of the plot or to do a lineto to that
% first point. Using this command this action can be set to a lineto.
```

```
%
```

```
% Example:
```

```
%
```

```
% \pgfsetlinetofirstplotpoint
```

```
\def\pgfsetlinetofirstplotpoint{\let\pgf@plot@first@action=\pgfpathlineto}
```

```
% Sets the action taken for the first point of a plot to a moveto.
```

```
%
```

```
% Example:
```

```
%
```

```
% \pgfsetmovetofirstplotpoint
```

```
\def\pgfsetmovetofirstplotpoint{\let\pgf@plot@first@action=\pgfpathmoveto}
```

```
\let\pgf@plot@first@action=\pgfpathmoveto
```

```
%  
% Handlers  
%
```

```
% This handler converts each plot stream command into a lineto  
% command, except for the first, which is converted to the action that  
% has previously been specified using \pgfsetlinetofirstplotpoint or  
% \pgfsetmovetofirstplotpoint.
```

```
%  
% Example:  
%  
% \pgfplotshandlerlineto  
% \pgfplotxyfile{mytable}
```

```
\def\pgfplotshandlerlineto{%  
  \def\pgf@plotstreamstart{%  
    \global\let\pgf@plotstreampoint=\pgf@plot@line@handler%  
    \global\let\pgf@plotstreamspecial=\pgfutil@gobble%  
    \global\let\pgf@plotstreamend=\relax%  
  }%  
}
```

```
\def\pgf@plot@line@handler#1{%  
  \pgf@plot@first@action{#1}%  
  \global\let\pgf@plotstreampoint=\pgfpathlineto%  
}
```

```
% This handler turns creates a series of lineto commands, with the  
% last command being a closepath, resulting in a closed path.
```

```
%  
% Example:  
%  
% \pgfplotshandlerpolygon  
% \pgfplotxyfile{mytable}
```

```
\def\pgfplotshandlerpolygon{%  
  \def\pgf@plotstreamstart{%  
    \global\let\pgf@plotstreampoint=\pgf@plot@line@handler@close%  
    \global\let\pgf@plotstreamspecial=\pgfutil@gobble%  
    \global\let\pgf@plotstreamend=\pgfpathclose%  
  }%  
}
```

```
}
```

```
\def\pgf@plot@line@handler@close#1{%  
  \pgfpathmoveto{#1}%  
  \global\let\pgf@plotstreampoint=\pgfpathlineto%  
}
```

```
% More handlers are defined in pgflibraryplohandlers
```

```
% This handler discards the plot.
```

```
%
```

```
% Example:
```

```
%
```

```
% \pgfplothandlerdiscard
```

```
% \pgfplotxyfile{mytable}
```

```
\def\pgfplothandlerdiscard{%  
  \def\pgf@plotstreamstart{%  
    \global\let\pgf@plotstreampoint=\pgfutil@gobble%  
    \global\let\pgf@plotstreamspecial=\pgfutil@gobble%  
    \global\let\pgf@plotstreamend=\relax%  
  }%  
}
```

```
% This handler records each plot stream command to a macro. This is  
% useful if plot commands are difficult to generate and need to be  
% ``recycled" later on.
```

```
%
```

```
% Example:
```

```
%
```

```
% \pgfplothandlerrecord{\myplot}
```

```
% \pgfplotxyfile{mytable} % stored in \myplot now
```

```
% \pgfplothandlerline
```

```
% \myplot
```

```
% \pgftransformxshift{1cm}
```

```
% \myplot
```

```

\def\pgfplotstreamrecord#1{%
  \def\pgf@plot@recordto{#1}%
  \gdef#1{\pgfplotstreamstart}%
  \def\pgf@plotstreamstart{%

\gdef\pgf@plotstreampoint####1{\expandafter\gdef\expandafter#1\expandafter{#1\pgfplotstream
point{####1}}}%

\gdef\pgf@plotstreamspecial####1{\expandafter\gdef\expandafter#1\expandafter{#1\pgfplotstrea
mspecial{####1}}}%

\gdef\pgf@plotstreamend{\expandafter\gdef\expandafter#1\expandafter{#1\pgfplotstreamend}}%
}%
}

```

```

% Read a plot stream from a file and plot it.
%
% #1 = file from which to read things
%
% File format:
%
% Each line of the file should begin with two numbers separated by a
% space. Such a line with number #1 and #2 is converted to a
% \pgfplotstreampoint{\pgfpointxy{#1}{#2}}. Extra characters following
% on the line are ignored.
%
% Lines starting with ``%" and ``#" are ignored.
%
% Example:
%
% \pgfplotxyfile{tableformgnuplot.dat}

```

```

\def\pgfplotxyfile#1{%
  \begingroup%
  \pgfplotstreamstart%
  \openin\r@pgf@reada=#1
  \ifeof\r@pgf@reada
    \PackageWarning{pgf}{Plot data file `#1' not found.}
  \else
    \catcode`\#=14
    \pgf@readxyfile%
  \fi
}

```

```

\pgfplotstreamend%
\endgroup%
}

\let\pgf@savdpar=\par%
\def\pgf@partext{\par}
\def\pgf@readxyfile{%
\pgfutil@read\r@pgf@reada to \pgf@temp%
\let\par=\pgf@savdpar%
\edef\pgf@temp{\pgf@temp}%
\ifx\pgf@temp\pgfutil@empty%
\else\ifx\pgf@temp\pgf@partext%
\else%
\expandafter\pgf@parsexyline\pgf@temp\pgf@stop%
\fi\fi%
\ifeof\r@pgf@reada\else\expandafter\pgf@readxyfile\fi%
}

\def\pgf@parsexyline#1 #2 #3\pgf@stop{%
\pgfplotstreampoint{\pgfpointxy{#1}{#2}}%
}

```

```

% Read a plot stream from a file and plot it.
%
% #1 = file from which to read things
%
% File format:
%
% Like xy, except that each line contains three numbers, which are
% converted to xyz coordiantes.
%
% Example:
%
% \pgfplotxyfile{tableformgnuplot.dat}

```

```

\def\pgfplotxyzfile#1{%
\beginpgfgroup%
\pgfplotstreamstart%
\openin\r@pgf@reada=#1
\ifeof\r@pgf@reada
\PackageWarning{pgf}{Plot data file `#1' not found.}

```

```

\else
  \catcode`\#=14
  \pgf@readxyzfile%
\fi
\pgfplotstreamend%
\endgroup%
}

\def\pgf@readxyzfile{%
  \pgfutil@read\r@pgf@reada to \pgf@temp%
  \ifx\pgf@temp\pgfutil@empty%
  \else\ifx\pgf@temp\pgf@partext%
  \else%
    \expandafter\pgf@parsexyzline\pgf@temp\pgf@stop%
  \fi\fi%
  \ifeof\r@pgf@reada\else\expandafter\pgf@readxyzfile\fi%
}

\def\pgf@parsexyzline#1 #2 #3 #4\pgf@stop{%
  \pgfplotstreampoint{\pgfpointxyz{#1}{#2}{#3}}%
}

```

```

% Render a function using gnuplot.
%
% #1 = filename prefix for .gnuplot and .table files (optional,
%   default is \jobname)
% #2 = gnuplot function text
%
% Description:
%
% This command will write a file called #1.gnuplot that sets up
% some gnuplot commands to write output to a file called
% #1.table. Then it calls gnuplot (using the \write18 mechanism)
% to execute the file. Then it reads #2.table using \pgfplotxyfile.
%
% Example:
%
% \pgfplothandlerlineto
% \pgfplotgnuplot[\jobname]{plot [x=0:5] x*sin(x)}
{
  \catcode`\%=12

```

```

\catcode\"=12
\def\pgf@gnuplot@head{set table \noexpand\pgf@plottablefile@quoted; set format "%.5f"}
}

\let\pgf@plotwrite=\w@pgf@writea
\newif\ifpgf@resample@plot

\def\pgfplotgnuplot{\pgfutil@ifnextchar[{\pgf@plotgnuplot}{\pgf@plotgnuplot[\jobname]}}%
\def\pgf@plotgnuplot[#1]#2{%
  \pgf@resample@plottrue%
  \pgfutilpreparefilename{#1.gnuplot}%
  \let\pgf@plotgnuplotfile=\pgfretval
  \pgfutilpreparefilename{#1.table}%
  \let\pgf@plottablefile=\pgfretval
  \let\pgf@plottablefile@quoted=\pgfretvalquoted
  % Check, whether it is up-to-date
  \openin\pgfutil@inputcheck=\pgf@plotgnuplotfile\relax
  \ifeof\pgfutil@inputcheck%
  \else%
    \pgfutil@read\pgfutil@inputcheck to\pgf@temp% ignored
    \pgfutil@read\pgfutil@inputcheck to\pgf@plot@line%
    \closein\pgfutil@inputcheck
  \edef\pgf@plot@code{#2\space}%
  \ifx\pgf@plot@code\pgf@plot@line%
    \openin\pgfutil@inputcheck=\pgfretval\relax
    \ifeof\pgfutil@inputcheck%
    \else%
      \closein\pgfutil@inputcheck
      \pgf@resample@plotfalse%
    \fi%
  \fi%
\fi
\ifpgf@resample@plot%
  \immediate\openout\pgf@plotwrite=\pgf@plotgnuplotfile\relax
  \immediate\pgfutil@write\pgf@plotwrite{\pgf@gnuplot@head}%
  \immediate\pgfutil@write\pgf@plotwrite{#2}%
  \immediate\closeout\pgf@plotwrite%
  \immediate\pgfutil@write18{gnuplot \pgf@plotgnuplotfile}%
\fi%
% temporarily redefine \pgf@parsexyline
\let\pgf@savdparsexyline=\pgf@parsexyline%
\let\pgf@parsexyline=\pgf@parsegnuplotxyline%
\pgfplotxyfile{\pgf@plottablefile}%
\let\pgf@parsexyline=\pgf@savdparsexyline%
}

```



```

\def\pgf@parsegnuplotxyline#1 #2 #3\pgf@stop{%
  \edef\pgf@xyline@flag@val{#3}%
  \edef\pgf@xyline@flag@undef{u\space}%
  \ifx\pgf@xyline@flag@val\pgf@xyline@flag@undef%
    \else%
      \pgfplotstreampoint{\pgfpointxy{#1}{#2}}%
    \fi%
}

```

% This producer handler plots a function using pgf's mathematical engine.

%

% #1 = variable

% #2 = domain for the variable

% #3 = point, typically defined in terms of the value of the variable

%

% Description:

%

% This producer will iterate the variable #1 over all variables in #2

% (using the \foreach statement). For each value, a plot coordinate

% #3 is created.

%

% Note that this command is pretty slow.

%

% Example:

%

% \pgfplotstreamlineto

% \pgfplotfunction{x}{0,0.1,...,3.141}{\pgfpointxy{x}{sin(x)}}

```

\def\pgfplotfunction#1#2#3{%
  \pgfplotstreamstart%
  \foreach#1in{#2}%
  {%
    \pgf@process{#3}%

```

```

  \pgfplotstreamstart%

```

```

  \foreach#1in{#2}%

```

```

  {%

```

```

    \pgf@process{#3}%

```

```

\edef\pgf@marshal{\noexpand\pgfplotstreampoint{\noexpand\pgfqpoint{\the\pgf@x}{\the\pgf@y}
}}%

```

```

  \pgf@marshal%

```

```

}

```

```

\pgfplotstreamend%

```

```

}

```

\endinput