The todonotes package*

Henrik Skov Midtiby
henrikmidtiby@gmail.com

December 7, 2008

Abstract

The `todonotes` package allows you to insert to–do items in your document. At any point in the document a list of all the inserted to–do items can be listed with the `\listoftodos` command.

Contents

*This document corresponds to todonotes .dtx, dated 2008/12/07.

1 Introduction

The `todonotes` package makes three commands available to the user: `\todo[]{}`, `\missingfigure{}` and `\listoftodos`. `\todo[]{}` and `\missingfigure{}` makes it possible to insert notes in your document about things that has to be done later (todonotes . . .). I developed the basic functionality of the package while I worked on my bachelor project.

1.1 Usage

`\todo` My most common usage of the todonotes package, is to insert an uncustomized todonotes somewhere in a latex document. An example of this usage is the command

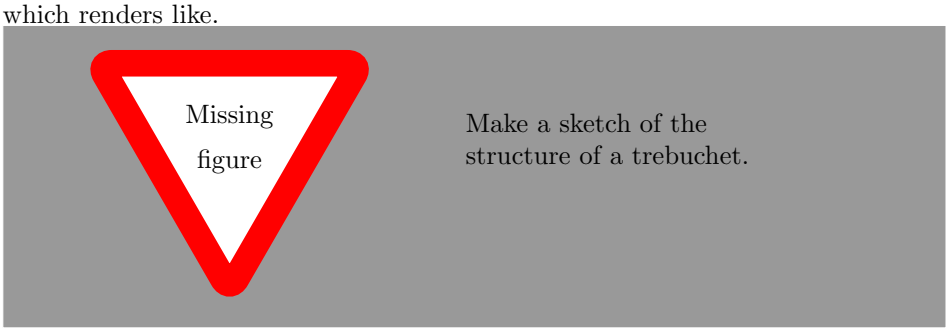
`\todo{Make a cake \ldots}`,

Make a cake . . . which renders like. The `\todo` command has this structure: `\todo[`⟨*options*⟩`]{`⟨*todo text*⟩`}`. The `todo text` is the text that will be shown in the todonote and in the list of todos. The optional argument `options`, allows the user to customize the appearance of the inserted todonotes. For a description of all the options see section 1.3.

`\missingfigure` The `\missingfigure` command inserts an image containing an attention sign and the given text. The command takes only one argument `\missingfigure{`⟨*text*⟩`}`, a text string that could describe what the figure should consist of. An example of its usage could be

`\missingfigure{Make a sketch of the structure of a trebuchet.}`

which renders like.



`\listoftodos` The `\listoftodos` command inserts a list of all the todos in the current document. `\listoftodos` takes no arguments. For this document the list of to–do's looks like.

Todo list

1.2 Package options

`disable` If the option `disable` is passed to the package, the macros usually defined by the package (`\todo`, `\listoftodos` and `\missingfigure`) are defined as macros with no effect, and thus all inserted notes are removed.

`danish` `german` Use the danish or german translations of the text strings "List of todos" and "Missing figure". The default is to use none of these options, which results in english text strings.

`colorinlistoftodos` Adds a small colored square in front of all items in the Todo list. The color of the square is the same as the fill color of the inserted todonote. This can be usefull if there are different types of todos (insert reference, explain in detail, ...) where the color of the inserted todonote marks the type of todo.

`color` `backgrouncolor` `linecolor` `bordercolor` These options sets the default colors for the todo command. There is three colors that can be specified. The border color (default `bordercolor=black`) around the inserted text, the color behind the inserted text (default `backgroundcolor=orange`) and the color of the line connecting the inserted textbox with the current location in the text (default `linecolor=orange`). Setting the `color` option to `val` passes this value on to the background and line color options. The specified colors must be valid according to the `xcolor` package.

`textwidth` `textwidth=length` sets the width of a todo item in the margin to `length`. The width of inline todonotes will allways be the same as the current line width.

`prependcaption` The `prependcaption` option triggers a special behaviour of the `caption=val` option for the todo command, where the given value `val` is inserted in the inserted todonote.

`dvistyle` When a document with todonotes is compiled with plain latex (to a dvi-file), there is an issue with the visual appearance[1]. The option `dvistyle` changes the appearance of the inserted todonotes to avoid this problem.

1.3 Options for the todo command

There are several options that can be given to the `\todo` command. All the options are described here and often I have included examples of the change in visual appearance.

`color` `backgroundcolor` `linecolor` `bordercolor` These options set the color that is used in the current todo command. The

[1] The problem is placement of text inside the colored boxes.

And a green note

color classes is the same as used in the color package options, see section 1.2. Default values can be set by the color options when the todonotes package is loaded. The todo notes inserted in this paragraph is created with the command `\todo[color=green!40]{And a green note}`. The color of the inserted note could be used to mark different types of tasks (insert references, explain something in detail, ...), this could be streamlined by defining new commands like below.

```
\newcommand{\insertref}[1]{\todo[color=green!40]{#1}}
\newcommand{\explainindetail}[1]{\todo[color=red!40]{#1}}
```

Anything but default colors

An example that uses all of the color options is given below .

```
\todo[linecolor=green!70!white, backgroundcolor=blue!20!white,
bordercolor=red]{Anything but default colors}.
```

`line / noline`

A note with no line connecting it to the placement in the original text.

If you want to get rid of the line connecting the inserted note with the place in the text where the note occurs in the latex code, the option `noline` can be used.
`\todo[noline]{A note with no line ...}`

`inline / noinline`

It is possible to place a todonote inside the text instead of placing it in the margin, this could be desireable if the text in the note has a considerable length.
`\todo[inline]{A todonote placed in the text}`

A todonote placed in the text

Another usage for the inline option is when you want to add a todonote to a figure caption.

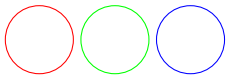


Figure 1: A text explaining the image.
Fill those circles ...

```
\begin{wrapfigure}{r}[20mm]{40mm}
\begin{tikzpicture}
\draw[red] (0, 0) circle(0.45);
\draw[green] (1, 0) circle(0.45);
\draw[blue] (2, 0) circle(0.45);
\end{tikzpicture}
\caption{A text explaining the image.
\todo[inline]{Fill those circles \ldots}}
\end{wrapfigure}
```

`size`

A note with a large fontsize.

`size=val` changes the size of the text inside the todonote. The commands used to create the notes below are
`\todo[size=\Large]{A note with a large fontsize.}` and
`\todo[inline, size=\tiny]{Note with very small fontsize.}`.

Note with very small fontsize.

`list / nolist`

When the option `nolist` is given, the todo item will not appear in the list of todos.

`caption`

The `caption` option enables the user to specify a short description of the

Short note: A very long and tedious note that cannot be on one line in the list of todos.

todonote that are inserted in the list of todos instead of the full todonote text.

```
\todo[caption={Short note}]{A very long and tedious note that
cannot be on one line in the list of todos.}.
```

The effect of this option is altered with the package option `prependcaption`.

1.4 Options for the missingfigure command

Currently the missingfigure command takes no optional arguments.

1.5 Options for the listoftodos command

Currently the listoftodos command takes no optional arguments.

1.6 Known issues

1.6.1 Package loading order

The todonotes package requires the following packages.

- ifthen
- xkeyval
- hyperref
- xcolor
- tikz
- calc
- graphicx (is loaded via the tikz package)

When todonotes are loaded in the preamble, the package checks if these packages all are loaded. If that is not the case it loads the missing packages with no options given. If you want to give some specific options to some of these packages, you have to load them *before* the todonotes package, otherwise you will get an "Option clash" error when latex works on the document.

1.6.2 Wrapping of long lines in list of todos

When a document is compiled with latex (and not pdflatex) long items in the list of todos are not wrapped into several lines, and do instead continue to the right out of the page.

1.7 Things to improve

This is a list of things I consider to improve sometime in the future. I havent been done yet as I lack the time or skills to implement them. Patches with implementations of these tasks will be appreciated and might be included in the package if it will improve the package quality.

1.7.1 Owner information

Option for the todo command.

```
\todo[owner={Fabrice}]{Stuff}
```

Add info on who "owns" the current todo. Idea: Fabrice Niessen

1.7.2 Due date

Option for the todo command.

```
\todo[due=2008-12-07]{Stuff}
```

Add info on when the current todo is due. Might be enhanced by a time line of the todos that have a due date assigned. Idea: Fabrice Niessen

1.7.3 Mark accomplished todos

```
\todo[done]{Stuff}
```

Idea: Fabrice Niessen

1.7.4 Enumeration of the todo notes

Add counters to the inserted notes. Idea: Henrik

1.8 Usage methods

In this section I have collected some different methods to use the `todonotes` package.

1.8.1 Define new commands with fixed options

Often there is a need for marking different classes of things to do (add reference, rewrite, ...). One way to do this, is to define some new commands as shown below (idea from Florent B.).

```
\newcommand{\addref}{\todo[color=red!40]{Add reference.}}
\newcommand{\rewrite}[1]{\todo[color=green!40]{#1}}
```

To distinguish between the different types of todos, the `todonotes` package can be loaded with the `colorinlistoftodos` option, which adds small colored squares to the list of todos.

```
\usepackage[colorinlistoftodos]{todonotes}
```

1.8.2 Combination with the fixme package

Thomas Arildsen has mailed me this use case. Check the documentation for the `fixme` package, as the code below relies directly on it (the `\FDUser` command is augumented when `\begin{document}` is reached).

```
\usepackage[user,nomargin]{fixme}
\usepackage{todonotes}
\newcommand{\FXUser}[2]{\todo[inline,size=\small]{{\bfseries #1:} #2}}
```

1.8.3 Altering the linespacing of todonotes

The `setspace` package lets you alter the line spacing of smaller sections of your document. The primary construct is the `spacing` environment, which is demonstrated below.

```
\begin{spacing}{0.5}
Some lines with a decreased line spacing. This is accomplished
using the setspace package that is included in standard latex
distributions.
\end{spacing}
```

Some lines with a decreased line spacing. This is accomplished using the setspace package that is included in standard latex distributions.

Some lines with a decreased line spacing. This is accomplished using the setspace package that is included in standard latex distributions.: Some lines with a decreased line spacing. This is accomplished using the setspace package that is included in standard latex distributions.

Using the `spacing` environment we can define a new todonote command using the code below:

```
\newcommand{\smalltodo}[2][]
    {\todo[caption={#2}, #1]
    {\begin{spacing}{0.5}#2\end{spacing}}}
```

Todonotes with decreased line spacing can now be inserted as follows

```
\smalltodo[size=\footnotesize]{
Some lines with a decreased line spacing. This is accomplished
using the setspace package that is included in standard latex
distributions.}
```

1.8.4 Comments "a la Word"

Fabrice Niessen sent me the following usecase. The idea is to define a new command `\mycomment` which adds a counter and optionally the initials of the auther to the inserted todonote.

```
\newcounter{mycomment}
\newcommand{\mycomment}[2][]{%
   % initials of the author (optional) + note in the margin
```

```
\refstepcounter{mycomment}%
{%
    \setstretch{0.7}% spacing
    \todo[color={red!100!green!33},size=\small]{%
        \textbf{Comment [\uppercase{#1}\themycomment]:}~#2}%
}}
```

Comment [HSM1]: Testing first time.: **Comment [HSM1]:** Testing first time.

Comment [HSM2]: Testing second time.: **Comment [HSM2]:** Testing second time.

The command `\mycomment[HSM]{Testing first time.}` is displayed like shown in the left margin, and another call of the command is added below `\mycomment[HSM]{Testing second time.}`.

2 Implementation

Identifies the package and loads the packages dependences.

```
\ProvidesPackage{todonotes}[2008/12/07]
\RequirePackage{ifthen}
\RequirePackage{xkeyval}
\RequirePackage{hyperref}
\RequirePackage{xcolor}
\RequirePackage{tikz}
\RequirePackage{calc}
```

Some default values are set

```
\newcommand{\@todonotes@text}{}%
\newcommand{\@todonotes@backgroundcolor}{orange}
\newcommand{\@todonotes@linecolor}{orange}
\newcommand{\@todonotes@bordercolor}{black}
\newcommand{\@todonotes@textwidth}{\marginparwidth}
```

2.1 Declaration of options for the package

In this part the various options for the package are defined.

Define the default text strings and set localization options for the danish and german languages.

```
\newcommand{\@todonotes@todolistname}{Todo list}
\newcommand{\@todonotes@MissingFigureUp}{Missing}
\newcommand{\@todonotes@MissingFigureDown}{figure}
\newcommand{\@todonotes@SetTodoListName}[1]{\renewcommand{\todolistname}{#1}}
\newcommand{\@todonotes@SetMissingFigureUp}[1]{\renewcommand{\MissingFigureUp}{#1}}
\providecommand{\SetMissingFigureDown}[1]{\renewcommand{\MissingFigureDown}{#1}}
\DeclareOptionX{danish}{%
    \@todonotes@SetTodoListName{G\o{}rem\aa{}lsliste}%
    \@todonotes@SetMissingFigureUp{Manglende}%
    \@todonotes@SetMissingFigureDown{figur}%
}
\DeclareOptionX{german}{%
    \@todonotes@SetTodoListName{Liste der noch zu erledigenden Punkte}%
```

```
    \@todonotes@SetMissingFigureUp{Fehlende}%
    \@todonotes@SetMissingFigureDown{Abbildung}%
}
```

Make it possible to disable the functionality of the package. If this option is given, the commands `\todo{}` and `\listoftodos` are defined as commands with no effect. (But you can still compile you document with these commands).

```
\newif{\if@todonotes@disabled}
\DeclareOptionX{disable}{\@todonotes@disabledtrue}
```

Show small boxes in the list of todos with the color of the inserted todonotes.

```
\newif{\if@todonotes@colorinlistoftodos}
\DeclareOptionX{colorinlistoftodos}{\@todonotes@colorinlistoftodostrue}
```

The default style behaves bad when compiled with latex (some text placement problems). The dvistyle option, changes the visual behavior to avoid this text placement problem.

```
\newif{\if@todonotes@dviStyle}
\DeclareOptionX{dvistyle}{\@todonotes@dviStyletrue}
```

Create a color option.

```
\define@key{todonotes.sty}%
    {color}{
        \renewcommand{\@todonotes@backgroundcolor}{#1}
        \renewcommand{\@todonotes@linegroundcolor}{#1}}
```

Make the background color of the notes as an option.

```
\define@key{todonotes.sty}%
    {backgroundcolor}{\renewcommand{\@todonotes@backgroundcolor}{#1}}
```

Make the line color of the notes as an option.

```
\define@key{todonotes.sty}%
    {linecolor}{\renewcommand{\@todonotes@linecolor}{#1}}
```

Make the color of the notes box color as an option.

```
\define@key{todonotes.sty}%
    {bordercolor}{\renewcommand{\@todonotes@bordercolor}{#1}}
```

Set whether short captions given as arguments to the todo command should be included in the inserted todonote.

```
\newif{\if@todonotes@prependcaption}
\@todonotes@prependcaptionfalse
\DeclareOptionX{prependcaption}{\@todonotes@prependcaptiontrue}
```

Make the text width as an option.

```
\define@key{todonotes.sty}%
    {textwidth}{\renewcommand{\@todonotes@textwidth}{#1}}
```

Finally process the given options.

```
\ProcessOptionsX
```

2.2 Options for the todo command

In this part the various options for commands in the package are defined. Set an arbitrarily fill color

```
\newcommand{\@todonotes@currentlinecolor}{}%
\newcommand{\@todonotes@currentbackgroundcolor}{}%
\newcommand{\@todonotes@currentbordercolor}{}%
\define@key{todonotes}{color}{%
    \renewcommand{\@todonotes@currentlinecolor}{#1}%
    \renewcommand{\@todonotes@currentbackgroundcolor}{#1}}%
\define@key{todonotes}{linecolor}{%
    \renewcommand{\@todonotes@currentlinecolor}{#1}}%
\define@key{todonotes}{backgroundcolor}{%
    \renewcommand{\@todonotes@currentbackgroundcolor}{#1}}%
\define@key{todonotes}{bordercolor}{%
    \renewcommand{\@todonotes@currentbordercolor}{#1}}%
```

Set a relative font size

```
\newcommand{\@todonotes@sizecommand}{}%
\define@key{todonotes}{size}{\renewcommand{\@todonotes@sizecommand}{#1}}%
```

Should the todo item be included in the list of todos?

```
\newif\if@todonotes@appendtolistoftodos%
\define@key{todonotes}{list}[]{\@todonotes@appendtolistoftodostrue}%
\define@key{todonotes}{nolist}[]{\@todonotes@appendtolistoftodosfalse}%
```

Should the todo item be displayed inline?

```
\newif\if@todonotes@inlinenote%
\define@key{todonotes}{inline}[]{\@todonotes@inlinenotetrue}%
\define@key{todonotes}{noinline}[]{\@todonotes@inlinenotefalse}%
```

Should the note in the margin be connected to the insertion point in the text?

```
\newif\if@todonotes@line%
\define@key{todonotes}{line}[]{\@todonotes@linetrue}%
\define@key{todonotes}{noline}[]{\@todonotes@linefalse}%
```

Should the text in the list of todos be different from the text in the todonote?

```
\newcommand{\@todonotes@caption}{}%
\newif\if@todonotes@captiongiven%
\define@key{todonotes}{caption}%
    {\renewcommand{\@todonotes@caption}{#1}%
    \@todonotes@captiongiventrue}%
```

Preset values of the options

```
\presetkeys%
    {todonotes}%
    {linecolor=\@todonotes@linecolor,%
    backgroundcolor=\@todonotes@backgroundcolor,%
    bordercolor=\@todonotes@bordercolor,%
    noinline,%
    line, list, size=\normalsize}{}%
```

2.3 The main code part

Here is the actual macros defined. If the option "disable" was passed to the package define empty commands.

```
\if@todonotes@disabled%
    \newcommand{\listoftodos}{}
    \newcommand{\todo}[2][]{\ignotespaces}
    \newcommand{\missingfigure}[1]{}
\else % \if@todonotes@disabled
```

Define the `\listoftodos` command and define the appearance of the list of todos.

```
\newcommand{\listoftodos}
    {\section*{\@todonotes@todolistname} \@starttoc{tdo}}
\newcommand{\l@todo}
    {\@dottedtocline{1}{0em}{2.3em}}
```

Define styles used by the todo command

```
\tikzstyle{notestyle} = [
    draw=\@todonotes@currentbordercolor,
    fill=\@todonotes@currentbackgroundcolor,
    line width=0.5pt,
    text width = \@todonotes@textwidth - 1.6 ex - 1pt,
    inner sep = 0.8 ex]
\tikzstyle{notestyleleft} = [
    notestyle,
    left]
\tikzstyle{connectstyle} = [
    thick,
    draw=\@todonotes@currentlinecolor]
\tikzstyle{inlinenotestyle} = [
    notestyle,
    text width=\linewidth - 1.6 ex - 1 pt]
```

`\todo` Define the todo command

```
\newcommand{\todo}[2][]{%
\renewcommand{\@todonotes@text}{#2}%
\renewcommand{\@todonotes@caption}{#2}%
\setkeys{todonotes}{#1}%
```

Add the item to the list of todos. When the option `colorinlistoftodos` is given to the package a small colored square is added in front of the text.

```
\if@todonotes@appendtolistoftodos%
    \phantomsection%
    \if@todonotes@captiongiven%
    \else%
        \renewcommand{\@todonotes@caption}{#2}%
    \fi%
    \if@todonotes@colorinlistoftodos%
        \addcontentsline{tdo}{todo}{\protect{%
            \colorbox{\@todonotes@currentbackgroundcolor}%
                {\textcolor{\@todonotes@currentbackgroundcolor}{\tiny i}}%
```

```
            \@todonotes@caption}}%
    \else%
        \addcontentsline{tdo}{todo}{\protect{\@todonotes@caption}}%
    \fi%
\fi%
```

Prepend the short caption given if it is requested

```
\if@todonotes@captiongiven
    \if@todonotes@prependcaption
        \renewcommand{\@todonotes@text}{\@todonotes@caption: #2}
    \fi
\fi
```

Place the todonote as indicated by the options (inline or in a marginpar), below is the code for the inline placement.

```
\if@todonotes@inlinenote%
    \if@todonotes@dviStyle%
        {\par\noindent\begin{tikzpicture}[remember picture]%
            \draw node[inlinenotestyle] {};\end{tikzpicture}\par}%
        {\noindent \@todonotes@sizecommand \@todonotes@text}%
        {\par\noindent\begin{tikzpicture}[remember picture]%
            \draw node[inlinenotestyle] {};\end{tikzpicture}\par}%
    \else%
        {\par\noindent\begin{tikzpicture}[remember picture]%
            \draw node[inlinenotestyle] {\@todonotes@sizecommand
\@todonotes@text};%
            \end{tikzpicture}\par}%
    \fi%
\else%
```

When the todonote should be placed inside a marginpar, the code below is applied. First is the current location in the document stored, this enables us later to connect this point with the inserted todonote.

```
\begin{tikzpicture}[remember picture, baseline=-0.75ex]%
    \node [coordinate] (inText) {};%
\end{tikzpicture}%
\marginpar[{% Draw note in left margin
\if@todonotes@dviStyle%
    \begin{tikzpicture}[remember picture]%
        \draw node[notestyle] {};%
    \end{tikzpicture}\\ %
    \begin{minipage}{\@todonotes@textwidth}%
    \@todonotes@sizecommand \@todonotes@text%
    \end{minipage}\\%
    \begin{tikzpicture}[remember picture]%
        \draw node[notestyle] (inNote) {};%
    \end{tikzpicture}%
\else%
    \begin{tikzpicture}[remember picture]%
        \draw node[notestyle] (inNote)%
            {\@todonotes@sizecommand \@todonotes@text};%
```

```
    \end{tikzpicture}%
\fi%
\if@todonotes@line%
    \begin{tikzpicture}[remember picture, overlay]%
        \draw[connectstyle]%
        ([yshift=-0.2cm] inText)%
        -| ([xshift=0.2cm] inNote.east)%
        -| (inNote.east);%
    \end{tikzpicture}%
\fi%
```

In the book documentclass (which is a twoside layout), the `\marginpar` marginpar macro takes two arguments `\marginpar[left]{right}`. If both arguments are given, latex will decide in which side the margin note has to be inserted, and then use the corresponding commands.

```
}]{% Draw note in right margin
\if@todonotes@dviStyle%
    \begin{tikzpicture}[remember picture]%
        \draw node[notestyle] {};%
    \end{tikzpicture}\\%
    \begin{minipage}{\@todonotes@textwidth}%
    \@todonotes@sizecommand \@todonotes@text%
    \end{minipage}\\%
    \begin{tikzpicture}[remember picture]%
        \draw node[notestyle] (inNote) {};%
    \end{tikzpicture}%
\else%
    \begin{tikzpicture}[remember picture]%
        \draw node[notestyle](inNote)%
            {\@todonotes@sizecommand \@todonotes@text};%
    \end{tikzpicture}%
\fi%
\if@todonotes@line%
    \begin{tikzpicture}[remember picture, overlay]%
        \draw[connectstyle]%
        ([yshift=-0.2cm] inText)%
        -| ([xshift=-0.2cm] inNote.west)%
        -| (inNote.west);%
    \end{tikzpicture}%
\fi%
}%
\fi%
\ignorespaces%
}%
```

`\missingfigure` Defines the `\missingfigure` macro.

```
\newcommand{\missingfigure}[1]{
\addcontentsline{tdo}{todo}{Figure: \protect{#1}}%
\par
\noindent
```

```
\begin{tikzpicture}
\draw[fill=black!40, draw = white, line width=0pt]
    (-3, -2.5) rectangle +(\textwidth, 4cm);
\draw (3, 0) node[right, text width=4cm] {#1};
\draw[red, fill=white, rounded corners = 5pt, line width=10pt]
    (30:2cm) -- (150:2cm) -- (270:2cm) -- cycle;
\draw (0, 0.3) node {\@todonotes@MissingFigureUp};
\draw (0, -0.3) node {\@todonotes@MissingFigureDown};
\end{tikzpicture}
}% Ending \missingfigure command
\fi % Ending \@todonotes@ifdisabled
```

Change History