

The `todonotes` package*

Henrik Skov Midtiby
`henrikmidtiby@gmail.com`

August 18, 2020

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

1	Introduction	2
1.1	Usage	2
1.2	Package options	4
1.3	Options for the <code>todo</code> command	5
1.4	Options for the <code>missingfigure</code> command	8
1.5	Options for the <code>listoftodos</code> command	9
1.6	Known issues	9
1.7	Things to improve	12
1.8	Usage methods	13
2	Implementation	19
2.1	Declaration of options for the package	19
2.2	Options for the <code>todo</code> command	23
2.3	The main code part	26

*This document corresponds to `todonotes` v1.1.3, dated 2020/08/18.

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.

Some alternatives for the `todonotes` package are:

- [easy-todo](#)
Depends on `color`, `tocloft` and `ifthen`, small feature set.
- [fixmetodonotes](#)
Depends on `graphicx`, `color`, `transparent`, `watermark`, `fix-cm`, `ulem` and `tocloft`, small feature set.
- [todo](#)
Depends on `amssymb`, medium feature set.
- [fixme](#)
Large package with a lot of features.

The main reason for considering other packages is that the `todonotes` package is quite large and relies heavily on `tikz`. This can slow down compilation of large documents significantly. The mentioned alternatives have a different feature set and does not rely on `tikz`, which makes them require fewer resources.

1.1 Usage

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

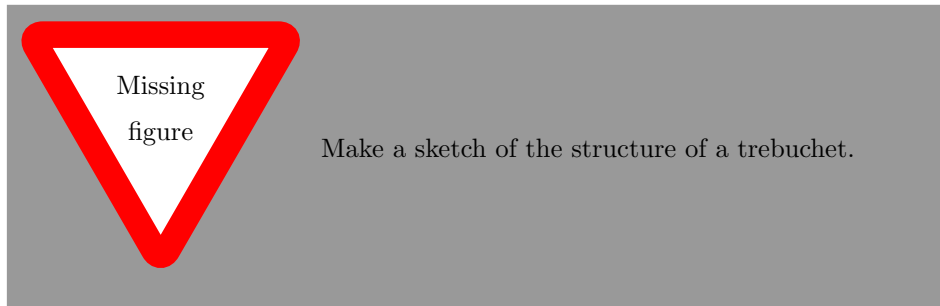
```
\todo{Make a cake \ldots},
```

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.

Make a cake . . .























`\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

 Make a cake ...	2
Figure: Make a sketch of the structure of a trebuchet.	2
 And a green note	5
 Anything but default colors	5
 A note without a shadow	6
 A note with a shadow	6
 Test of option.	6
 A note with no line connecting it to the placement in the original text.	6
 A todonote placed in the text	6
 Fill those circles ...	6
 A note with a large font size.	6
 Note with very small font size.	6
 Short note	6
 Short note with prepend	7
 Short note with noprepnd	7
 Testing.	7
 Testing author option.	7
 Testing author option.	7
 Testing the option inlinewidth.	7
 Testing the option inlinepar.	7
Figure: Testing a long text string	8
Figure: Testing a long text string	8
Figure: Add an image ...	8
Figure: Testing	9
Figure: Testing figcolor	9
 Test of newly defined command.	13
 Test of newly defined command, requesting a green color.	13
 1: A numbered todonote.	14
 2: Another numbered todonote.	14

	Comment [HSM1]: Testing first time.	14
	Comment [HSM2]: Testing second time.	14
	Some lines with a decreased line spacing. This is accomplished using the setspace package that is included in standard latex distributions. . . .	15
	Some lines with a decreased line spacing. This is accomplished without using any special packages.	15
	2do	16
	Translation	16
	Translation	16
	1.8.9. A numbered todo.	17
	1. Small notes with links back to the list of todos.	17
	1. Smart notes with links back to the list of todos.	17
	fix text	18

`\todototoc` The `\todototoc` command adds an entry to the table of contents for list of todos. The command should be placed right before the `\listoftodos` command.

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.

`obeyDraft`, `obeyFinal` When the option `obeyDraft` is given, the package checks if the one of the options `draft`, `draftcls` or `draftclsnofoot` is given (this option is usually given to the documentclass). If the `draft` option is given, the functionality of the package is enabled and otherwise the effect of the package is disabled. The option `obeyFinal` does something similar, except that the `todonotes` package is only disabled if the `final` option is given.

`danish`, `german`, `ngerman`,
`english`, `french`, `swedish`
`spanish`, `catalan`, `italian`
`portuguese`, `dutch`,
`croatian`
`colorinlistoftodos` Use 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. Currently the following languages are supported: catalan, croatian, danish, dutch, english, french, german, ngerman, italian, portuguese, spanish and swedish.

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

`tickmarkheight` `tickmarkheight=length` set the height of the tickmark at the location where

	the todo is inserted, the default value is <code>Opt</code> .
<code>textwidth</code>	<code>textwidth=length</code> sets the width of a todo item in the margin to <code>length</code> . The width of inline todonotes will always be the same as the current line width.
<code>textsize</code>	<code>textsize=value</code> sets the default text size of the inserted todonotes to the given value. Value is the "name" of the used font size, eg. if the desired fontsize is <code>\tiny</code> use <code>textsize=tiny</code> . The default value is <code>textsize=normalsize</code> .
<code>prependcaption</code>	The <code>prependcaption</code> option triggers a special behaviour of the <code>caption=val</code> option for the todo command, where the given value <code>val</code> is inserted in the inserted todonote.
<code>shadow</code>	If the <code>shadow</code> option is given, the inserted todonotes will be displayed with a gray shadow. I expect that the option will trigger problems with tikz versions prior to 2.0.
<code>dvistyle</code>	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 <code>dvistyle</code> changes the appearance of the inserted todonotes to avoid this problem.
<code>figwidth</code>	The <code>figwidth=length</code> option sets the default width of the figure inserted by the <code>\missingfigure</code> command. The default value is <code>\linewidth</code> .

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. Default values for these options can be set using the `setuptodonotes` command.

```
\setuptodonotes{fancyline, color=blue!30}
```

<code>disable</code>	The <code>disable</code> option can be given directly to the todo command. If given the command has no effect.
<code>color</code>	These options set the color that is used in the current todo command. The color classes is the same as used in the color package options, see section 1.2.
<code>backgroundcolor</code>	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 <code>\todo[color=green!40]{And a green note}</code> . 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.
<code>textcolor</code>	
<code>linecolor</code>	
<code>bordercolor</code>	

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

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

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

shadow / noshadow

If the package is loaded with the `shadow` option, a shadow is added to all notes, otherwise no shadows are shown. This global setting can be overwritten by the `shadow` and `noshadow` options to the `todo` command. To insert a note with and without shadows, use the `shadow` and `noshadow` options as shown here:

A note without a shadow

A note with a shadow

```
\todo[shadow]{A note with a shadow}
\todo[noshadow]{A note without a shadow}
```

tickmarkheight

Test of option.

`tickmarkheight=length` set the height of the tickmark at the location where the `todo` is inserted. An example of the option in use is shown here.

```
\todo[tickmarkheight=0.1cm]{Test of option.}
```

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 desirable 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.

```
\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}
```

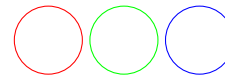


Figure 1: A text explaining the image.

Fill those circles ...

size

`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 font size.} and
\todo[inline, size=\tiny]{Note with very small font size.}
```

A note with a large font size.

Note with very small font size.

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

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

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

```
\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` or the `prepend / noprepend` option for the `todo` command.

`prepend / noprepend`

The options `prepend` and `noprepend` can be used for setting whether a given caption should be prepended to the `todonote` or not. Globally this can be set using the `prependcaption` option for the package. Below is the effect of the option shown using the code:

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

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

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

The `fancyline` option inserts a curved arrow, pointing from the inserted note to the insertion point. The option is used like this:

`fancyline`

Testing.

```
\todo[fancyline]{Testing.}
```

`author`

The `author` option takes a parameter, the name of the author. The given name is inserted in the `todonote`.

Xavier

Testing author option.

```
Xavier: Testing author option.
```

```
\todo[author=Xavier]{Testing author option.}  
\todo[author=Xavier, inline]{Testing author option.}
```

`inlinewidth`

The `inlinewidth` option makes it possible to alter the width of inline todos.

Testing the option inlinewidth.

```
\todo[inline, inlinewidth=5cm]{Testing the option inlinewidth.}
```

`inlinepar / noinlinepar`

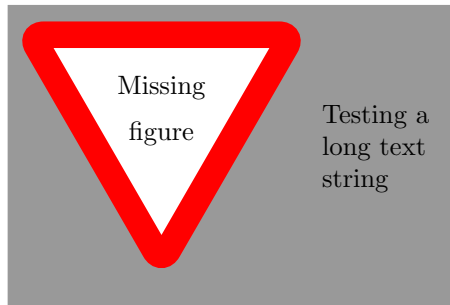
The options `inlinepar` and `noinlinepar` (default) determine whether the inserted inline `todonote` is surrounded with `par` macros that enforce new lines prior to and after inserted inline `todonotes`. Testing the option `inlinepar`. This is a line after the inserted `todonote`.

```
\todo[inline, inlinewidth=5cm, noinlinepar]{Testing the option inlinepar.}
```

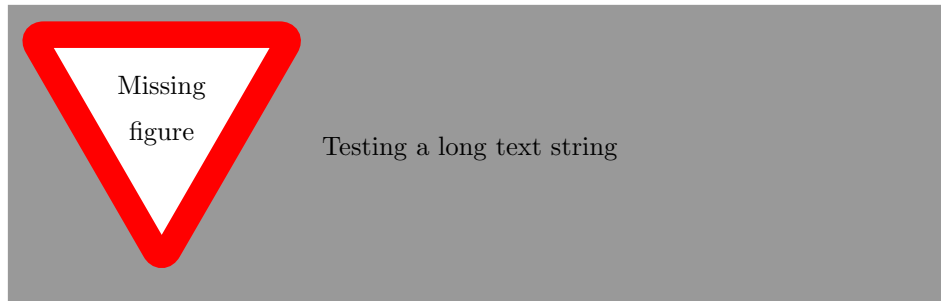
1.4 Options for the missingfigure command

figwidth The `figwidth=length` option sets the width of the figure inserted by the `\missingfigure` command. Length values below *6cm* might trigger some problems with the visual appearance. Try to compare the default of the missing figure command, when the option is given or not.

```
\missingfigure[figwidth=6cm]{Testing a long text string}
```

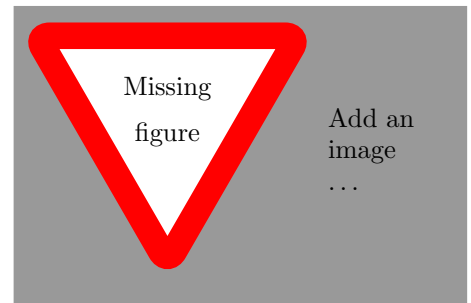


```
\missingfigure{Testing a long text string}
```



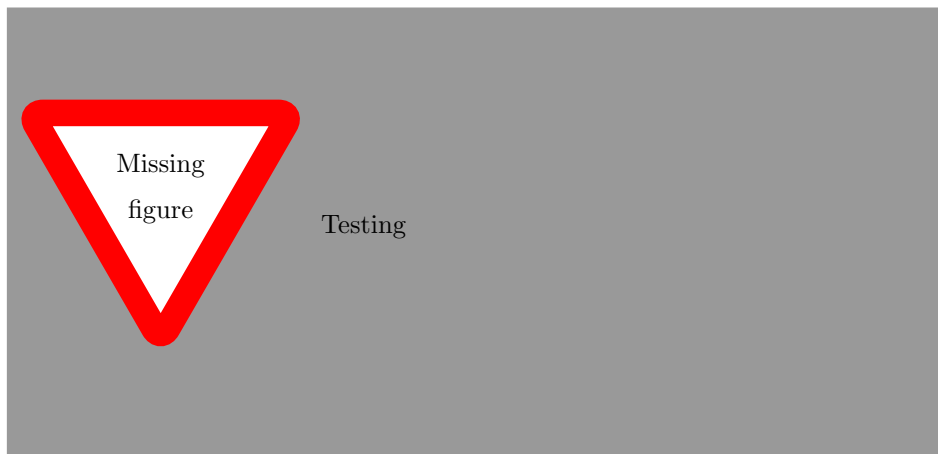
Another usage of the option is when `\missingfigure` is used in the `wrapfigure` environment.

```
\begin{wrapfigure}[1]{r}[2cm]{6cm}  
\missingfigure[figwidth=6cm]{Add an image \ldots}  
\end{wrapfigure}
```



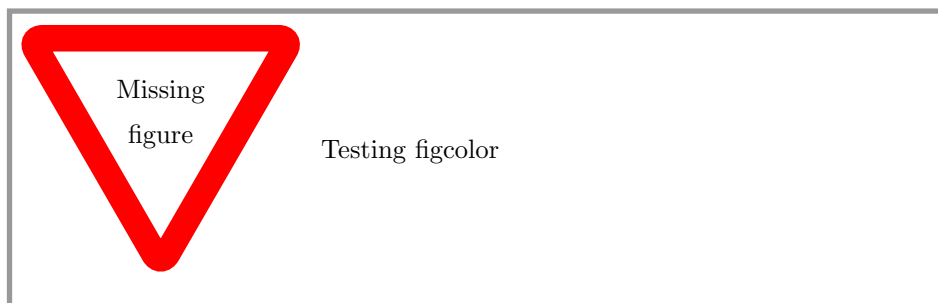
figheight The `figheight=length` option changes the height of the inserted missing figure. The default height is *4cm* and using values lower than this might cause the warning sign to pop out of the gray area.


```
\missingfigure[figheight=6cm]{Testing a long text string}
```



`figcolor` The `figcolor=color` options sets the background color of inserted missing figures. The default color is `black!40`.

```
\missingfigure[figcolor=white]{Testing figcolor}
```



1.5 Options for the `listoftodos` command

The `\listoftodos` command takes one optional argument, that defines the name of the inserted list of todos.

```
\listoftodos[I can be called anything]
```

1.6 Known issues

1.6.1 Package loading order

The `todonotes` package requires the following packages.

- ifthen
- xkeyval
- 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.

If both the menukeys and the xcolor (with the option `table`) package should be loaded, the following order must be used.

```
\usepackage[table]{xcolor}
\usepackage{todonotes}
\usepackage{menukeys}
```

The following packages must be loaded before the todonotes package:

- polyglossia

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.

1.6.3 Conflicts with the amsart documentclass

The `amsart` document class redefines some internal commands that is used by the todonotes package, this will cause an malfunctioning `\listoftodos` command. The following code to circumvent the problem was given by Dan Luecking on `comp.text.tex`

```
\makeatletter
\providecommand\@dotsep{5}
\makeatother
\listoftodos\relax
```

NOT TESTED NOT TESTED NOT TESTED

Dominique suggests the following workaround.

```
\makeatletter
\providecommand\@dotsep{5}
\def\listtodoname{List of Todos}
\def\listoftodos{\@starttoc{tdo}\listtodoname}
\makeatother
```

1.6.4 Unknown option "remember picture"

If latex throws the error

```
Package tikz Error: I do not know what to do with the option ‘remember picture’.
```

It probably means that your latex installation is outdated, as only newer versions of latex driver for tikz supports the `remember picture` option. For additional info consult "Section 9.2.2 Producing PDF Output" in the tikz manual. <http://www.ctan.org/tex-archive/graphics/pgf/base/doc/generic/pgf/pgfmanual.pdf>

1.6.5 Todonotes wrongly placed in the margin

When using some document classes or packages, the todonotes inserted in the page margin can be placed quite oddly. This is often caused by a wrong value of the `\marginparwidth` lenght. Try using the code below in your preamble to see if this cures the problem.

```
\setlength{\marginparwidth}{2cm}
```

If the todonotes are inserted in the wrong margin, the solution is the `\reversemarginpar` command. When this command is issued the following marginpars (which todonotes relies on) is inserted in the other margin.

1.6.6 Reduce number of warnings

If the width of the inserted todonotes is forced to be larger than the available space in the margin, a lot of warnings are issued. This can be reduced with the following code.

```
\usepackage[textwidth=3.7cm]{todonotes}  
\setlength{\marginparwidth}{3.7cm}
```

1.6.7 Todonotes in footnotes

Placing todonotes in footnotes throws a lot of errors. Richard Stanton comes with the following work around.

```
\renewcommand{\marginpar}{\marginnote}
```

1.6.8 Trouble with classicthesis.sty

[Problem description on tex.stackexchange.com.](#)

Solution by Stefan Kottwitz. The problem is caused by the redefinition of `\marginpar` in `classicthesis.sty`. `\marginpar` is used by todonotes. It can be fixed by restoring the original meaning, if you don't need the `classicthesis` marginpar style. Just add this to your document preamble: `\let\marginpar\oldmarginpar`

1.6.9 todos in math and floating environments

It is not possible to insert `todonotes` into math environments or floating environments like `figure` or `table`. By replacing the `marginpar` command with a `marginnote` this can be enabled. For todos inside math environments the `todo` must be placed in an `mbox` or `\textrm` command.

```
\documentclass{article}
\usepackage{todonotes}
\usepackage{marginnote}
\let\marginpar\marginnote

\begin{document}
\begin{equation}
a^2=b^2+c^2 \textrm{\todo{Test}}
\end{equation}
\begin{figure}
\centering
\caption{Caption\todo{tests}.}
\label{fig}
\end{figure}
```

1.7 Things to improve

This is a list of things I consider to improve sometime in the future. It have not 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

1.8 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 Define new commands with arbitrary default options

If you do not like the default values of the standard `todo` command, it is possible to define a new command with the similar functionality of `\todo` with custom default values.

```
\newcommand{\todoredefined}[2] []
{\todo[color=red, #1]{#2}}
```

The new command can now be used like shown below

```
\todoredefined{Test of newly defined command.}
\todoredefined[color=green]{Test of newly defined command, requesting a green color.}
```

This can be done with all the accepted options for the `\todo` command.

1.8.3 Enumerate todonotes

If the inserted todonotes should be enumerated, it is possible to define a new command with the desired behaviour.

```
\newcounter{todocounter}
\newcommand{\todonum}[2] []
{\stepcounter{todocounter}\todo[#1]{\thetodocounter: #2}}
```

Test of newly defined command.

Test of newly defined command, requesting a green color.

1: A numbered todonote.

2: Another numbered todonote.

The idea is to define a new counter `todocounter`, and insert the value of the counter in each todonote. The new command can be used like

```
\todonum{A numbered todonote.}
\todonum{Another numbered todonote.}
```

1.8.4 Comments "a la Word"

Fabrice Niessen sent me the following use case. The idea is to define a new command `\mycomment` which adds a counter and optionally the initials of the author 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 [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.}`.

1.8.5 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 augmented when `\begin{document}` is reached).

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

1.8.6 Altering the line spacing 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.

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}}}
```

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

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.}
```

A different approach is given by Vitaly.

```
\newcommand{\tinytodo}[2] []
  {\todo[caption={#2}, size=\small, #1]{\renewcommand{\baselinestretch}{0.5}\selectfont#2\par}}
```

Some lines with a decreased line spacing. This is accomplished without using any special packages.

It looks like seen here.

```
\tinytodo{
Some lines with a decreased line spacing. This is accomplished
without using any special packages.}
```

1.8.7 Marking new / old sections

Sometimes a whole section has to be marked by some means. You might want to try the following.

```
\todo[inline, caption={Some text}]{
\begin{minipage}{\linewidth}
Some text that might differ from the text given to the caption
option.
\end{minipage}
}
```

It is important to add the caption={text} option, otherwise latex will try to embed a minipage in the table of contents which triggers an error. Inside the minipage environment almost anything could be placed, except for other todo commands.

To streamline use the following command was suggested by Stefan Pinnow.

```
\newcommand\todoIn[2] []{\todo[inline, caption={2do}, #1]{
\begin{minipage}{\textwidth-4pt}#2\end{minipage}}}
```

This example renders like

```
\todoin{
Some text.
\begin{align}
\sin(\theta)^2 + \cos(\theta)^2 = 1
\end{align}
A formula and a list
\begin{itemize}
\item An item
\end{itemize}
}
```

Some text.

$$\sin(\theta)^2 + \cos(\theta)^2 = 1 \tag{1}$$

A formula and a list

- An item

1.8.8 Link to list of todos

Using the `hyperref` package it is possible to add a link from the inserted todonotes to the list of todos. The example were supplied by Andreas Plank.

```
% Define a counter for the inserted todonotes.
\newcounter{todoListItems}
\newcommand{\todoTrans}[2][ ]{
  % Increment counter
  \addtocounter{todoListItems}{1}
  \todo[%
    caption={\protect\hypertarget{todo\thetodoListItems}{Translation},
    #1]
  {
    #2 \hfill
    \hyperlink{todo\thetodoListItems}{\uparrow}
  }
}
```

The idea behind the code is to embed a `hypertarget` in each entry in the list of todos. In the todonotes a link to the entry in the list of todos is inserted as an arrow that points upwards. Using the `\todoTrans` command like below, the following two notes have been inserted.

```
\todoTrans{papersflyver}
\todoTrans[inline]{damplokomotiv}
```

papersflyver



damplokomotiv



1.8.9 Numbered todonotes

The inserted todonotes can be argumented with the current subsubsection number. The code is shown below.

```
\newcommand{\ntodo}[2] [] {\todo[#1]{\thesubsubsection{ }. #2}}
```

By changing `\thesubsubsection` to `\thesection`, the current section number can be inserted instead of the subsubsection number. The result looks like. Which were generated by the code

```
\ntodo{A subsection numbered todo.}.
```

1.8.9. A numbered todo.

1.8.10 Combining several modifications

Manduca have combined several of the modifications above into a highly specialized todo command. She uses the code:

```
\newcounter{todoListItems}
\newcommand{\sstodo}[2] []
{\addtocounter{todoListItems}{1}
\todo[caption={\protect\hypertarget{todo\thetodoListItems}{ }\thesection. #2}, #1]
{\begin{spacing}{1} \hfill \hyperlink{todo\thetodoListItems}{#2} \end{spacing} }}
```

Small notes with links back to the list of todos.

Smart notes with links back to the list of todos.

Using this approach it is possible to customize the behavior of the inserted notes to a very high degree.

1.8.11 Alter the appearance of the list of todos

Marco Daniel gives the following example of how to add section numbers to the elements in the list of todos. The code is slightly modified from <http://tex.stackexchange.com/questions/18838/replacing-page-number-with-other-counter-in-listoftodos>. An example of the modified list of todos is shown below, the complete code example is given in the example directory.

```
This is a note . . . . . see 1.1 at p. 2
This is another note . . . . . see 1.2 at p. 4
```

1.8.12 Tikz externalization issues

Using the tikz externalization framework together with todonotes can lead to some problems. One solution is to disable the externalization just before the `todo` command is issued and then reactivate externalization afterwards. The `ruggedtodo` handles this deactivation and reactivation.

```
\usetikzlibrary{external}
\tikzexternalize
\newcommand{\ruggedtodo}[2] [] {\tikzexternaldisable\todo[#1]{#2}\tikzexternalenable}
```

1.8.13 Highlight text to fix

Tobias Winchen provides the following example on how to highlight text related to the inserted todonote. Example `wrong text` continues here. Notice that the code relies on the soul package.

```
\makeatletter
\if@todonotes@disabled
\newcommand{\hlfix}[2]{#1}
\else
\newcommand{\hlfix}[2]{\textl{#1}\todo{#2}}
\fi
\makeatother
Example \hlfix{wrong text}{fix text}~continues here.
```

fix text

2 Implementation

Identifies the package and loads the packages dependences.

```
1 \ProvidesPackage{todonotes}[2018/11/22]
2 \RequirePackage{ifthen}
3 \RequirePackage{xkeyval}
4 \RequirePackage{xcolor}
5 \RequirePackage{tikz}
6 \usetikzlibrary{positioning}
7 \usetikzlibrary{shadows}
8 \RequirePackage{calc}
```

Implement a function for setting up the todonotes package.

```
9 \newcommand\setuptodonotes[1]{\presetkeys{todonotes}{#1}{}}
```

Some default values are set

```
10 \newcommand{\@todonotes@text}{}%
11 \newcommand{\@todonotes@backgroundcolor}{orange}
12 \newcommand{\@todonotes@textcolor}{black}
13 \newcommand{\@todonotes@linecolor}{orange}
14 \newcommand{\@todonotes@bordercolor}{black}
15 \newcommand{\@todonotes@tickmarkheight}{0cm}
16 \newcommand{\@todonotes@textwidth}{\marginparwidth}
17 \newcommand{\@todonotes@textsize}{\normalsize}
18 \newcommand{\@todonotes@figwidth}{\linewidth}
19 \newcommand{\@todonotes@figheight}{4cm}
20 \newcommand{\@todonotes@figcolor}{black!40}

21 \AtBeginDocument{
22 \ifx\undefined\phantomsection
23 \newcommand{\phantomsection}{}
24 \fi
25 \ifdim \marginparwidth < 2cm
26 \PackageWarning{todonotes}{The length marginparwidth is
27 less than 2cm and will most likely cause issues with the
28 appearance of inserted todonotes.
29 The issue can be solved by adding a line like
30 \setlength{\marginparwidth}{2cm}
31 prior to loading the todonotes package.} \else\fi%
32 }
```

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.

```
33 \newcommand{\@todonotes@todolistname}{Todo list}
34 \newcommand{\@todonotes@MissingFigureText}{Figure}
35 \newcommand{\@todonotes@MissingFigureUp}{Missing}
36 \newcommand{\@todonotes@MissingFigureDown}{figure}
```

```

37 \newcommand{\@todonotes@SetTodoListName}[1]
38   {\renewcommand{\@todonotes@todolistname}{#1}}
39 \newcommand{\@todonotes@SetMissingFigureText}[1]
40   {\renewcommand{\@todonotes@MissingFigureText}{#1}}
41 \newcommand{\@todonotes@SetMissingFigureUp}[1]
42   {\renewcommand{\@todonotes@MissingFigureUp}{#1}}
43 \newcommand{\@todonotes@SetMissingFigureDown}[1]
44   {\renewcommand{\@todonotes@MissingFigureDown}{#1}}
45 \newif{\if@todonotes@reverseMissingFigureTriangle}
46 \DeclareOptionX{catalan}{%
47   \@todonotes@SetTodoListName{Llista de feines pendants}%
48   \@todonotes@SetMissingFigureText{Figura}%
49   \@todonotes@SetMissingFigureUp{Figura}%
50   \@todonotes@SetMissingFigureDown{pendent}%
51 }
52 \DeclareOptionX{croatian}{%
53   \@todonotes@SetTodoListName{Popis obveza}%
54   \@todonotes@SetMissingFigureText{Slika}%
55   \@todonotes@SetMissingFigureUp{Nedostaje}%
56   \@todonotes@SetMissingFigureDown{slika}%
57 }
58 \DeclareOptionX{danish}{%
59   \@todonotes@SetTodoListName{G\o{}rem\aa{}lsliste}%
60   \@todonotes@SetMissingFigureText{Figur}%
61   \@todonotes@SetMissingFigureUp{Manglende}%
62   \@todonotes@SetMissingFigureDown{figur}%
63 }
64 \DeclareOptionX{dutch}{%
65   \@todonotes@SetTodoListName{Lijst van onafgewerkte taken}%
66   \@todonotes@SetMissingFigureText{Figuur}%
67   \@todonotes@SetMissingFigureUp{Ontbrekende}%
68   \@todonotes@SetMissingFigureDown{figuur}%
69 }
70 \DeclareOptionX{english}{%
71   \@todonotes@SetTodoListName{Todo list}%
72   \@todonotes@SetMissingFigureText{Figure}%
73   \@todonotes@SetMissingFigureUp{Missing}%
74   \@todonotes@SetMissingFigureDown{figure}%
75 }
76 \DeclareOptionX{french}{%
77   \@todonotes@SetTodoListName{Liste des points \`a traiter}%
78   \@todonotes@SetMissingFigureText{Figure}%
79   \@todonotes@SetMissingFigureUp{Figure}%
80   \@todonotes@SetMissingFigureDown{manquante}%
81   \@todonotes@reverseMissingFigureTrianglefalse
82 }
83 \DeclareOptionX{german}{%
84   \@todonotes@SetTodoListName{Liste der noch zu erledigenden Punkte}%
85   \@todonotes@SetMissingFigureText{Abbildung}%
86   \@todonotes@SetMissingFigureUp{Fehlende}%

```

```

87 \@todonotes@SetMissingFigureDown{Abbildung}%
88 }
89 \DeclareOptionX{italian}{
90 \@todonotes@SetTodoListName{Elenco delle cose da fare}%
91 \@todonotes@SetMissingFigureText{Figura}%
92 \@todonotes@SetMissingFigureUp{Figura}%
93 \@todonotes@SetMissingFigureDown{mancante}%
94 }
95 \DeclareOptionX{ngerman}{%
96 \@todonotes@SetTodoListName{Liste der noch zu erledigenden Punkte}%
97 \@todonotes@SetMissingFigureText{Abbildung}%
98 \@todonotes@SetMissingFigureUp{Fehlende}%
99 \@todonotes@SetMissingFigureDown{Abbildung}%
100 }
101 \DeclareOptionX{portuguese}{
102 \@todonotes@SetTodoListName{Lista de tarefas pendentes}%
103 \@todonotes@SetMissingFigureText{Figura}%
104 \@todonotes@SetMissingFigureUp{Figura}%
105 \@todonotes@SetMissingFigureDown{pendente}%
106 }
107 \DeclareOptionX{spanish}{
108 \@todonotes@SetTodoListName{Lista de tareas pendientes}%
109 \@todonotes@SetMissingFigureText{Figura}%
110 \@todonotes@SetMissingFigureUp{Figura}%
111 \@todonotes@SetMissingFigureDown{pendiente}%
112 }
113 \DeclareOptionX{swedish}{%
114 \@todonotes@SetTodoListName{Att g\{"o}ra-lista}%
115 \@todonotes@SetMissingFigureText{Figur}%
116 \@todonotes@SetMissingFigureUp{Figur}%
117 \@todonotes@SetMissingFigureDown{saknas}%
118 }

```

Create a counter, for storing the number of inserted todos.

```
119 \newcounter{@todonotes@numberoftodonotes}
```

Toggle whether the package should obey the global draft option.

```

120 \newif{\if@todonotes@obeyDraft}
121 \DeclareOptionX{obeyDraft}{\@todonotes@obeyDrafttrue}
122 \newif{\if@todonotes@isDraft}
123 \DeclareOptionX{draft}{\@todonotes@isDrafttrue}
124 \DeclareOptionX{draftcls}{\@todonotes@isDrafttrue}
125 \DeclareOptionX{draftclsnofoot}{\@todonotes@isDrafttrue}
126 \newif{\if@todonotes@obeyFinal}
127 \DeclareOptionX{obeyFinal}{\@todonotes@obeyFinaltrue}
128 \newif{\if@todonotes@isFinal}
129 \DeclareOptionX{final}{\@todonotes@isFinaltrue}

```

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).

```

130 \newif{\if@todonotes@disabled}
131 \DeclareOptionX{disable}{\@todonotes@disabledtrue}
    Show small boxes in the list of todos with the color of the inserted todonotes.
132 \newif{\if@todonotes@colorinlistoftodos}
133 \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.
134 \newif{\if@todonotes@dviStyle}
135 \DeclareOptionX{dvistyle}{\@todonotes@dviStyletrue}
    Create a color option.
136 \define@key{todonotes.sty}%
137   {color}{
138     \renewcommand{\@todonotes@backgroundcolor}{#1}
139     \renewcommand{\@todonotes@linecolor}{#1}}
    Make the background color of the notes as an option.
140 \define@key{todonotes.sty}%
141   {backgroundcolor}{\renewcommand{\@todonotes@backgroundcolor}{#1}}
    Make the text color of the notes as an option.
142 \define@key{todonotes.sty}%
143   {textcolor}{\renewcommand{\@todonotes@textcolor}{#1}}
    Make the line color of the notes as an option.
144 \define@key{todonotes.sty}%
145   {linecolor}{\renewcommand{\@todonotes@linecolor}{#1}}
    Make the color of the notes box color as an option.
146 \define@key{todonotes.sty}%
147   {bordercolor}{\renewcommand{\@todonotes@bordercolor}{#1}}
    Make the height of the line start marking as an option.
148 \define@key{todonotes.sty}%
149   {tickmarkheight}{\renewcommand{\@todonotes@tickmarkheight}{#1}}
    Set whether short captions given as arguments to the todo command should be
    included in the inserted todonote.
150 \newif{\if@todonotes@prependcaptionglobal}
151 \@todonotes@prependcaptionglobalfalse
152 \DeclareOptionX{prependcaption}{\@todonotes@prependcaptionglobaltrue}
    Make the text width as an option.
153 \define@key{todonotes.sty}%
154   {textwidth}{\renewcommand{\@todonotes@textwidth}{#1}}
    Make the text size as an option, accept both size and textsize.
155 \define@key{todonotes.sty}%
156   {textsize}{\renewcommand{\@todonotes@textsize}{#1}}
157 \define@key{todonotes.sty}%
158   {size}{\renewcommand{\@todonotes@textsize}{#1}}

```

Add option for shadows behind the inserted notes

```
159 \newcommand{\@todonotes@shadowenabledbydefault}{noshadow}
160 \DeclareOptionX{shadow}{\renewcommand{\@todonotes@shadowenabledbydefault}{shadow}}
```

Add option for the default width of the figure inserted with `\missingfigure`.

```
161 \define@key{todonotes.sty}%
162   {figwidth}{\renewcommand{\@todonotes@figwidth}{#1}}
163 \define@key{todonotes.sty}%
164   {figheight}{\renewcommand{\@todonotes@figheight}{#1}}
165 \define@key{todonotes.sty}%
166   {figcolor}{\renewcommand{\@todonotes@figcolor}{#1}}
```

Make the text width as an option.

```
167 % Finally process the given options.
168 %   \begin{macrocode}
169 \ProcessOptionsX*
```

If the `obeyDraft` is given, check whether one of the `draft`, `draftcls` or `draftclsnofoot` options are given and enable or disable the functionality of this package. If the `obeyFinal` option is given together with the `final` option the `todonotes` are disabled. The `disable` option will overrule the effect of `obeyDraft`.

```
170 \if@todonotes@disabled
171 \else
172 \if@todonotes@obeyDraft
173 \@todonotes@disabledtrue
174 \if@todonotes@isDraft
175 \@todonotes@disabledfalse
176 \fi
177 \fi
178 \if@todonotes@obeyFinal
179 \@todonotes@disabledfalse
180 \if@todonotes@isFinal
181 \@todonotes@disabledtrue
182 \fi
183 \fi
184 \fi
```

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

```
185 \gdef\@todonotes@currentlinecolor{\@todonotes@linecolor}%
186 \gdef\@todonotes@currentbackgroundcolor{\@todonotes@backgroundcolor}%
187 \gdef\@todonotes@currenttextcolor{\@todonotes@textcolor}%
188 \gdef\@todonotes@currentbordercolor{\@todonotes@bordercolor}%
189 \define@key{todonotes}{color}{%
190   \gdef\@todonotes@currentlinecolor{#1}%
191   \gdef\@todonotes@currentbackgroundcolor{#1}}%
192 \define@key{todonotes}{linecolor}{%
193   \gdef\@todonotes@currentlinecolor{#1}}%
```

```

194 \define@key{todonotes}{backgroundcolor}{%
195     \gdef\@todonotes@currentbackgroundcolor{#1}}%
196 \define@key{todonotes}{textcolor}{%
197     \gdef\@todonotes@currenttextcolor{#1}}%
198 \define@key{todonotes}{bordercolor}{%
199     \gdef\@todonotes@currentbordercolor{#1}}%

Toggle whether there is a shadow behind the inserted notes.
200 \newif\if@todonotes@useshadow%
201 \define@key{todonotes}{shadow}[]{\@todonotes@useshadowtrue}%
202 \define@key{todonotes}{noshadow}[]{\@todonotes@useshadowfalse}%

Define height of the inserted tickmark.
203 \define@key{todonotes}{tickmarkheight}{%
204     \renewcommand{\@todonotes@tickmarkheight}{#1}}%

Set a relative font size
205 \newcommand{\@todonotes@sizecommand}{}%
206 \define@key{todonotes}{size}{\renewcommand{\@todonotes@sizecommand}{#1}}%
207 }%

Should the todo item be disabled?
208 \newif\if@todonotes@localdisable%
209 \define@key{todonotes}{disable}[]{\@todonotes@localdisabletrue}%
210 \define@key{todonotes}{nodisable}[]{\@todonotes@localdisablefalse}%

Should the todo item be included in the list of todos?
211 \newif\if@todonotes@appendtolistoftodos%
212 \define@key{todonotes}{list}[]{\@todonotes@appendtolistoftodostrue}%
213 \define@key{todonotes}{nolist}[]{\@todonotes@appendtolistoftodosfalse}%

Should the todo item be displayed inline?
214 \newif\if@todonotes@inlinenote%
215 \define@key{todonotes}{inline}[]{\@todonotes@inlinenotetrue}%
216 \define@key{todonotes}{noinline}[]{\@todonotes@inlinenotefalse}%

217 \newif\if@todonotes@prependcaption%
218 \define@key{todonotes}{prepend}[]{\@todonotes@prependcaptiontrue}%
219 \define@key{todonotes}{noprepend}[]{\@todonotes@prependcaptionfalse}%

Should the note in the margin be connected to the insertion point in the text?
220 \newif\if@todonotes@line%
221 \define@key{todonotes}{line}[]{\@todonotes@linetrue}%
222 \define@key{todonotes}{noline}[]{\@todonotes@linefalse}%

Should the connection between note and insertion point be drawn in a fancy way?
(Does only work if line=true)
223 \newif\if@todonotes@fancyline\@todonotes@fancylinefalse%
224 \define@key{todonotes}{fancyline}[]{\@todonotes@fancylinetrue}%
225 \define@key{todonotes}{nofancyline}[]{\@todonotes@fancylinefalse}%

Author option.
226 \newcommand{\@todonotes@author}{%

```



```

227 \newif\if@todonotes@authorgiven%
228 \define@key{todonotes}{author}{%
229   \renewcommand{\@todonotes@author}{#1}%
230   \@todonotes@authorgiventruer}%
231 \define@key{todonotes}{noauthor}[]{\@todonotes@authorgivenfalse}%

```

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

```

232 \newcommand{\@todonotes@caption}{}%
233 \newif\if@todonotes@captiongiven%
234 \define@key{todonotes}{caption}{%
235   {\renewcommand{\@todonotes@caption}{#1}%
236   \@todonotes@captiongiventruer}%
237 \define@key{todonotes}{nocaption}[]{\@todonotes@captiongivenfalse}%

```

Change the current figure width and height.

```

238 \newcommand{\@todonotes@currentfigwidth}{\@todonotes@figwidth}
239 \define@key{todonotes}{%
240   {figwidth}{\renewcommand{\@todonotes@currentfigwidth}{#1-2pt}}
241 \newcommand{\@todonotes@currentfigheight}{\@todonotes@figheight}
242 \define@key{todonotes}{%
243   {figheight}{\renewcommand{\@todonotes@currentfigheight}{#1-2pt}}
244 \newcommand{\@todonotes@currentfigcolor}{\@todonotes@figcolor}
245 \define@key{todonotes}{%
246   {figcolor}{\renewcommand{\@todonotes@currentfigcolor}{#1}}

```

Change the width of an inline note.

```

247 \newcommand{\@todonotes@inlinewidth}{\linewidth}%
248 \define@key{todonotes}{%
249   {inlinewidth}{\renewcommand{\@todonotes@inlinewidth}{#1}}

```

Change if inline note is written to a separate line or not.

```

250 \newif\if@todonotes@inlinepar
251 \@todonotes@inlinepartrue
252 \define@key{todonotes}{inlinepar}[]{\@todonotes@inlinepartrue}%
253 \define@key{todonotes}{noinlinepar}[]{\@todonotes@inlineparfalse}%

```

Preset values of the options

```

254 \presetkeys%
255   {todonotes}%
256   {linecolor}=\@todonotes@linecolor,%
257   backgroundcolor=\@todonotes@backgroundcolor,%
258   textcolor=\@todonotes@textcolor,%
259   bordercolor=\@todonotes@bordercolor,%
260   tickmarkheight=0cm,%
261   nofancyline,%
262   nodisable,%
263   noinline,%
264   nocaption,%
265   noauthor,%
266   \@todonotes@shadowenabledbydefault,%
267   figwidth=\@todonotes@figwidth,%
268   figheight=\@todonotes@figheight,%

```

```

269   figcolor=\@todonotes@figcolor,%
270   line, list,%
271   inlinewidth=\linewidth,
272   inlinepar}{}%
273 \@temptokena\expandafter{\@todonotes@textsize}
274 \edef\next{\noexpand\presetkeys{todonotes}{size=\the\@temptokena}{}}
275 \next

```

2.3 The main code part

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

```

276 \if@todonotes@disabled%
277   \newcommand{\listoftodos}[1] [] {}
278   \newcommand{\@todo}[2] [] {}
279   \newcommand{\missingfigure}[2] [] {}
280 \else % \if@todonotes@disabled
    Define the \listoftodos command and define the appearance of the list of todos.
281 \newcommand{\listoftodos}[1] [\@todonotes@todolistname]
282   {\@ifundefined{chapter}{\section*{#1}}{\chapter*{#1}} \@starttoc{tdo}}
283 \newcommand{\l@todo}
284   {\@dottedtocline{1}{0em}{2.3em}}
    Define styles used by the todo command
285 \tikzstyle{notestylera} = [
286   draw=\@todonotes@currentbordercolor,
287   fill=\@todonotes@currentbackgroundcolor,
288   text=\@todonotes@currenttextcolor,
289   line width=0.5pt,
290   text width = \@todonotes@textwidth - 1.6 ex - 1pt,
291   inner sep = 0.8 ex,
292   rounded corners=4pt]

```

\@todo Define the \@todo command.

```

293 \newcommand{\@todo}[2] [] {}%
    Use the global value for determining the default prepend behavior.
294 \if@todonotes@prependcaptionglobal%
295 \@todonotes@prependcaptiontrue%
296 \else%
297 \@todonotes@prependcaptionfalse%
298 \fi%
    Store the original text for later usage and parse the given options.
299 \renewcommand{\@todonotes@text}{#2}%
300 \renewcommand{\@todonotes@caption}{#2}%
301 \setkeys{todonotes}{#1}%
    Add shadows to the inserted todonotes.
302 \if@todonotes@usesshadow%

```

```

303 \tikzstyle{notestyle} = [notestylera,%
304     general shadow={shadow xshift=0.5ex, shadow yshift=-0.5ex,%
305         opacity=1,fill=black!50}]%
306 \else%
307 \tikzstyle{notestyle} = [notestylera]%
308 \fi%

```

Update notestyles

```

309 \tikzstyle{notestyleleft} = [%
310     notestyle,%
311     left]%
312 \tikzstyle{connectstyle} = [%
313     thick,%
314     draw=\@todonotes@currentlinecolor]%
315 \tikzstyle{inlinenotestyle} = [%
316     notestyle,%
317     text width=\@todonotes@inlinewidth - 1.6 ex - 1 pt]%

```

If the option `disable` is given to the command, no output is generated.

```

318 \if@todonotes@localdisable%
319 \else%

```

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.

```

320 \addtocounter{@todonotes@numberoftodonotes}{1}%
321 \if@todonotes@appendtolistoftodos%
322     \phantomsection%
323     \if@todonotes@captiongiven%
324     \else%
325         \renewcommand{\@todonotes@caption}{#2}%
326     \fi%
327     \@todonotes@addElementToListOfTodos%
328 \fi%

```

Prepend the short caption given if it is requested

```

329 \if@todonotes@captiongiven%
330     \if@todonotes@prependcaption%
331         \renewcommand{\@todonotes@text}{\@todonotes@caption: #2}%
332     \fi%
333 \fi%

```

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

```

334 \if@todonotes@inlinenote%
335     \@todonotes@drawInlineNote%
336 \else%
337     \@todonotes@drawMarginNoteWithLine%
338 \fi%\if@todonotes@inlinenote
339 \fi%\if@todonotes@localdisable
340 }%

```

```

drawMarginNoteWithLine Define helper function drawMarginNoteWithLine.
341 \newcommand{\@todonotes@drawMarginNoteWithLine}{%
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.
342 \ifvmode
343   \vspace*{-\parskip}%   % backup if we are already in vertical mode
344   \vskip-\baselineskip % (and don't lose that space after a
345                       % pagebreak ...
346   \noindent
347 \fi
348 \begin{tikzpicture}[remember picture, overlay, baseline=-0.75ex]%
349   \node [coordinate] (inText) {};%
350 \end{tikzpicture}%
351 \marginpar[{} Draw note in left margin
352   \@todonotes@drawMarginNote%
353   \@todonotes@drawLineToLeftMargin%
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.
354 }]{% Draw note in right margin
355   \@todonotes@drawMarginNote%
356   \@todonotes@drawLineToRightMargin%
357 }%
358 }%

addElementToListOfTodos Define helper function addElementToListOfTodos.
359 \newcommand{\@todonotes@addElementToListOfTodos}{%
360   \if@todonotes@colorinlistoftodos%
361     \addcontentsline{tdo}{todo}{%
362       \fcolorbox{\@todonotes@currentbordercolor}%
363         {\@todonotes@currentbackgroundcolor}%
364         {\textcolor{\@todonotes@currentbackgroundcolor}{o}}%
365       \ \@todonotes@caption}%
366   \else%
367     \addcontentsline{tdo}{todo}{\@todonotes@caption}%
368   \fi}%

drawInlineNote Define helper function useSizeCommand.
369 \newcommand{\@todonotes@useSizeCommand}{%
370 \ifcsname \expandafter\string\@todonotes@sizecommand\endcsname
371 \csname \expandafter\string\@todonotes@sizecommand\endcsname%
372 \else
373 \@todonotes@sizecommand
374 \fi%
375 }%

```

drawInlineNote Define helper function drawInlineNote.

```
376 \newcommand{\@todonotes@drawInlineNote}{%
377   \if@todonotes@dviStyle%
378     {\if@todonotes@inlinepar\par\noindent\fi%
379       \begin{tikzpicture}[remember picture]%
380         \draw node[inlinenotestyle] {};
381       \end{tikzpicture}%
382     \if@todonotes@inlinepar\par\fi}%
383   \if@todonotes@authorgiven%
384     {\noindent \@todonotes@useSizeCommand \@todonotes@author:\,\@todonotes@text}%
385   \else%
386     {\noindent \@todonotes@useSizeCommand \@todonotes@text}%
387   \fi
388   {\if@todonotes@inlinepar\par\noindent\fi%
389     \begin{tikzpicture}[remember picture]%
390       \draw node[inlinenotestyle] {};
391     \end{tikzpicture}%
392   \if@todonotes@inlinepar\par\fi}%
393 \else%
394   {\if@todonotes@inlinepar\par\noindent\fi%
395     \begin{tikzpicture}[remember picture]%
396       \draw node[inlinenotestyle,font=\@todonotes@useSizeCommand] {%
397         \if@todonotes@authorgiven%
398           {\noindent \@todonotes@author:\,\@todonotes@text}%
399         \else%
400           {\noindent \@todonotes@text}%
401         \fi};%
402     \end{tikzpicture}%
403   \if@todonotes@inlinepar\par\fi}%
404 \fi}%
```

drawMarginNote Define helper function drawMarginNote.

```
405 \newcommand{\@todonotes@drawMarginNote}{%
406 \if@todonotes@dviStyle%
407   \begin{tikzpicture}[remember picture]%
408     \draw node[notestyle] {};%
409   \end{tikzpicture}\\%
410   \begin{minipage}{\@todonotes@textwidth}%
411     \if@todonotes@authorgiven%
412       \@todonotes@useSizeCommand \@todonotes@author \@todonotes@text%
413     \else%
414       \@todonotes@useSizeCommand \@todonotes@text%
415     \fi%
416   \end{minipage}\\%
417   \begin{tikzpicture}[remember picture]%
418     \draw node[notestyle] (inNote) {};%
419   \end{tikzpicture}%
420 \else%
421   \let\originalHbadness\hbadness%
422   \hbadness 100000%
```

```

423 \begin{tikzpicture}[remember picture,baseline=(X.base)]%
424 \node(X){\vphantom{\@todonotes@useSizeCommand X}};%
425 \if@todonotes@authorgiven%
426 \draw node[notestyle,font=\@todonotes@useSizeCommand,anchor=north] (inNote) at (X.n
427 {\@todonotes@author};%
428 \node(Y)[below=of X]{};%
429 \draw node[notestyle,font=\@todonotes@useSizeCommand,anchor=north] (inNote) at (X.s
430 {\@todonotes@text};%
431 \else%
432 \draw node[notestyle,font=\@todonotes@useSizeCommand,anchor=north] (inNote) at (X.n
433 {\@todonotes@text};%
434 \fi%
435 \end{tikzpicture}%
436 \hbadness \originalHbadness%
437 \fi}%

```

drawLineToRightMargin Define helper function drawLineToRightMargin.

```

438 \newcommand{\@todonotes@drawLineToRightMargin}{%
439 \if@todonotes@line%
440 \if@todonotes@fancyline%
441 \tikz[remember picture,overlay]{%
442 \tikzstyle{both}=[line width=3pt, draw, opacity=0.15]%
443 \tikzstyle{line}=[shorten >=5pt, line cap=round]%
444 \tikzstyle{head}=[shorten >=-1pt, dash pattern=on 0pt off 1pt, ->]%
445 \foreach \s in {line,head}{%
446 \draw[both,\s]
447 (inNote.north west).. controls +(0:0) and +(90:1.5)..([yshift=1ex] inText);%
448 }%
449 }%
450 \else%
451 \begin{tikzpicture}[remember picture, overlay]%
452 \draw[connectstyle]%
453 ([yshift=-0.2cm + \@todonotes@tickmarkheight] inText)%
454 -| ([yshift=-0.2cm] inText)%
455 -| ([xshift=-0.2cm] inNote.west)%
456 -| (inNote.west);%
457 \end{tikzpicture}%
458 \fi%
459 \fi}%

```

drawLineToLeftMargin Define helper function drawLineToLeftMargin.

```

460 \newcommand{\@todonotes@drawLineToLeftMargin}{%
461 \if@todonotes@line%
462 \if@todonotes@fancyline%
463 \tikz[remember picture,overlay]{%
464 \tikzstyle{both}=[line width=3pt, draw, opacity=0.15]%
465 \tikzstyle{line}=[shorten >=5pt, line cap=round]%
466 \tikzstyle{head}=[shorten >=-1pt, dash pattern=on 0pt off 1pt,->]%
467 \foreach \s in {line,head}{%
468 \draw[both,\s]

```

```

469 (inNote.north east).. controls +(0:0) and +(90:1.5)..([yshift=1ex] inText);%
470 }%
471 }%
472 \else%
473 \begin{tikzpicture}[remember picture, overlay]%
474 \draw[connectstyle]%
475 ([yshift=-0.2cm + \@todonotes@tickmarkheight] inText)%
476 -| ([yshift=-0.2cm] inText)%
477 -| ([xshift=0.2cm] inNote.east)%
478 -| (inNote.east);%
479 \end{tikzpicture}%
480 \fi%
481 \fi}%

```

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

```

482 \newcommand{\missingfigure}[2] [] {%
483 \setkeys{todonotes}{#1}%
484 \addcontentsline{todo}{todo}{\@todonotes@MissingFigureText: #2}%
485 \par
486 \noindent
487 \hfill
488 \begin{tikzpicture}
489 \draw[fill=\@todonotes@currentfigcolor, draw = black!40, line width=2pt]
490 (-2, -0.5*\@todonotes@currentfigheight-0.5cm)
491 rectangle +(\@todonotes@currentfigwidth, \@todonotes@currentfigheight);
492 \draw (2, -0.5) node[right, text
493 width=\@todonotes@currentfigwidth-4.5cm, font=\@todonotes@useSizeCommand] {#2};
494 \draw[red, fill=white, rounded corners = 5pt, line width=10pt]
495 (30:2cm) -- (150:2cm) -- (270:2cm) -- cycle;
496 \draw (0, 0.3) node {\@todonotes@MissingFigureUp};
497 \draw (0, -0.3) node {\@todonotes@MissingFigureDown};
498 \end{tikzpicture}\hfill
499 \null\par
500 }% Ending \missingfigure command
501 \fi% Ending \@todonotes@ifdisabled

```

`\todotoc` Inserts a reference to the list of todos in the table of contents. If `chapter` is defined, `chapter` is used as level otherwise will `section` be used. The `\todotoc` command respects the `disable` option.

```

502 \newcommand{\todotoc}
503 {%
504 \if@todonotes@disabled
505 \else
506 \addcontentsline{toc}{\@ifundefined{chapter}{section}{chapter}}{\@todonotes@todolistname}%
507 \fi
508 }

```

`\todo` Define the `\todo` command as a redirection to `\@todo`.

```

509 \newcommand{\todo}[2] [] {%

```

```

510 % Needed to output any dangling \item of a noskip section (see #36):
511 \if@inlabel \leavevmode \fi
512 \if@noskipsec \leavevmode \fi
513 \if@todonotes@inlinepar
514   \ifhmode
515     \@bspback
516     \@todonotes@vmodedefalse
517   \else
518     \@savsf\@m
519     \@savsk\z@
520     \@todonotes@vmodetrue
521   \fi
522   {\@todo[#1]{#2}}%
523   \@esphack%
524   \if@todonotes@vmode \par \fi
525 \else%
526   \@todo[#1]{#2}%
527 \fi}
528 \newif\if@todonotes@vmode

```


Change History

0.1	General: The first version of the package	1	0.4.2	General: Fixed a bug with the disable option to the package. . .	1
0.2	General: Updated the option handling of the package	1	0.5	General: Created a dtx file containing both source code and documentation of the package	1
0.2.1	General: Slightly modified by Kjell Magne Fauske to support notes in the left margin (for documentstyle book).	1	0.5.1	General: Updated the documentation	1
0.2.2	General: Added a missingfigure command	1	0.5.2	General: Fixed a bug that prevented the usage of the option french for babel. Bug report by Thomas Braun.	1
0.2.3	General: Made a dependency on the calc package	1	0.6	General: Added the caption option to the todo command.	1
0.3	General: Delayed the requirements for the hyperref package untill begin document and added an optional argument to the todo command for adding inline todonotes (Idea from Patrick Toche)	1	0.6.1	General: Added a new usecase with decreased line spacing.	1
0.3.1	General: Added some options to the todo macro (Idea: Patrick Toche) and made the listoftodos point at the inserted todos and not only the current / previous section, subsection or figure using the phantomsection macro.	1	0.6.2	General: Added a usecase by Fabrice Niessen.	1
0.4	General: Modified the behaviour of the inline todonotes, to avoid empty lines around the inline todonotes.	1	0.7	General: Added language options on request from Peter Zimmermann.	1
0.4.1	General: Added the option colorinlistoftodos which inserts a small box with the used fillcolor of the todonotes in the list of todos.	1	0.7.1	General: Reworked the color options for both the whole package and the todo command. General code clean up. Added the prependcaption package option.	1
			0.7.2	General: Avoid to change the fontsize inside the list of todos, fixing a bug revealed by Vladimir Zhuravlev.	1
			0.7.3	General: The localization options (danish and german) and the disable options, were all flawed by naming inconsistencies that	

	made then break the package.	0.8.4
	This have been fixed.	1
0.7.4	General: Fixed a bug related to the caption option for the todo command. Introduced a counter of the number of inserted todos.	1
0.7.5	General: Fixed a typo in a macroname.	1
0.7.6	General: Added a textsize option for the package and the prepend / noprepnd option for the todo command.	1
0.8	General: Added three new translations french, spanish and catalan thanks to Richard Dominique and Joan Queralt. Improved the visual appearance of the inserted notes (rounded corners and optional shadows) with code from Joan Queralt. Found an untranslated textstring "Figure" in the source. Added a figwidth option to the missingfigure command, patch by Paul Ivanov.	1
0.8.1	General: Added a space between the colored square and the text in the list of todos. Added a new usecase for marking old / new sections. Made the name of listoftodos changeable.	1
0.8.2	General: Italian translation by Gustavo Cevolani. Removed the dependence on the hyperref package.	1
0.8.3	General: Added a use case for linking to the list of todos, idea from Andreas Plank. Introduced a package option for listening to the draft option given to the document class.	1
	General: Fixed a bug related to the obeyDraft option.	1
0.8.5	General: Added two new usecases (enumeration of inserted todonotes and how to set custom default values). Changed the order of the use case examples.	1
0.8.6	General: Added a portuguese translation by Og DeSouza.	1
0.8.7	General: Updated portuguese translation. Added a ngerman alias for the german translation suggested by Michael Niedermair.	1
0.8.8	General: Added a new usecase from Vitaly. Fixed a bug reported by Oscar Gustafsson. Explained why the placement of todonotes in the margin fails in certain custom document classes.	1
0.8.9	General: Added a dutch translation by Ruben Ruben Vermeersch.	1
0.9.0	General: Added a english option as suggested by Marco Berghoff.	1
0.9.1	General: Added the todototoc command by idea from Sven Augustin.	1
0.9.2	General: Use chapter (if available) for the list of todos heading.	1
0.9.3	General: Make an internal definition of the todo command, for easing redefinition of the command behaviour.	1
0.9.4	General: Make the disable option work on a local scale.	1

0.9.5	General: Code simplification by extracting functionality to smaller macros.	1	using the presetkeys command. Updated list of alternatives to the todonotes package. Draw borders around coloured boxes in the list of todos, patch by Ze Loff.	1
0.9.6	General: Give fontsize to TikZ. Align notes with line where note is set. Added new option fancyline. Patches by Benjamin Kellermann.	1	1.0.3	General: Added the option figcolor to missing figure, patch by Pascal Hebbeker. Added Croatian translation by Ican Kokan. Changed default with of missingfigure. Removed some underfull box warnings, solution by Ernst Blecha.
0.9.7	General: Updated documentation.	1	1.0.4	General: Restructured documentation and placed some examples in the doc/examples subdirectory.
0.9.8	General: Suppress warnings about underfull / overfull boxes generated by the inserted todonotes. Patch by Peter M Schuler.	1	1.0.5	General: Example of how to change the space eating behaviour added by Anselm Wagner. Reduced the width of missingfigure to avoid bad box warnings.
0.9.9	General: Added author option, implementation provided by Xavier Alameda-Pineda. Example of modifying the listoftodos removing some protect commands with no effect.	1	1.0.6	General: Changed how whitespaces are handled around inserted todos to better mimick macros like index (Suggestion by Frank Mittelbach). Fix missing character warnings using patch by Niels Anders Danielsson. Use the specified textsize for missingfigure, patch by Johannes Twittmann. Added new options for inline todonotes based on code from Daniel Krenn. Added warning message when marginparwidth is less than 2cm, as suggested by ErikBoesen.
1.0.0	General: Mention trouble with the classicthesis style. Refer to some alternatives to the package. Added todoin command as suggested by Stefan Pinnow. Described how to use tikz externalize with todonotes. Added obeyDraft and obeyFinal options.	1	1.0.7	General: Changed the warning about marginparwidth to a proper warning message.
1.0.1	General: Fix spacing issues reported by Jonathan Zachhuber and Brent Longborough. Added figheight option to the missingfigure command as suggested by Kim Albertsson.	1	1.0.8	General: Improved spacing
1.0.2	General: Added Swedish translation by Emil Lundberg. Added usecase by Tobias Winchen. Mentioned that default arguments can be set			

behaviour so it mimics the index macro, patch by Frank Mittelbach. Center graphics vertically in missingfigure, patch by Philipp Allgeuer. Added new option tickmarkheight as suggested by Richard Niland.	1	customizable height. Steamline which options that are accepted to size commands (both commands (backslash tiny) and text (tiny) are now supported).	1
1.1.0		1.1.1	
General: Many improvements to the user interface by Frank Mittelbach. Added the command setuptodonotes. Added textcolor as an option. Marked the insertion point of notes with tickmarks with a		General: Implemented shadow and noshow options for the todo command.	1
		1.1.2	
		General: Fix issue 36 and 37.	1
		1.1.3	
		General: Fix issue 48 and make colors used in the last todo globally available.	1