The \texttt{todonotes} package∗

Henrik Skov Midtiby
henrikmidtiby@gmail.com
January 31, 2023

Abstract

The \texttt{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 \texttt{listoftodos} command.

Contents

1 Introduction 2
1.1 Usage 2
1.2 Package options 4
1.3 Options for the \texttt{todo} command 5
1.4 Options for the \texttt{missingfigure} command 8
1.5 Options for the \texttt{listoftodos} command 9
1.6 Known issues 9
1.7 Things to improve 12
1.8 Usage methods 13

2 Implementation 20
2.1 Declaration of options for the package 20
2.2 Options for the \texttt{todo} command 24
2.3 The main code part 27

∗This document corresponds to \texttt{todonotes} v1.1.6, dated 2023/01/31.
1 Introduction

The todonotes package makes four 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 ...). The \todostyle command allows the user to define named custom styles as abbreviations for lists of other options that can be given to the \todo command. 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.

\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.
Make a sketch of the structure of a trebuchet.

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 ........................ 6
- 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 .......................................... 7
- Short note with prepend ............................ 7
- Short note with noprepend .......................... 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
- description ........................................... 11
- Test of newly defined command ...................... 14
- Test of newly defined command, requesting a green color .................................. 14
- A todo with user-defined style red .................. 14
- 1: A numbered todonote ................................ 14
- 2: Another numbered todonote ........................ 14
- Comment [HSM1]: Testing first time ............... 15
- Comment [HSM2]: Testing second time ............. 15
Some lines with a decreased line spacing. This is accomplished using the \setspace package that is included in standard latex distributions. \ldots 15

Some lines with a decreased line spacing. This is accomplished without using any special packages. \ldots \ldots 15

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

\todostyle{\langle name\rangle}{\langle style\rangle}  The \texttt{\todostyle} command defines a new optional argument to the \texttt{\todo} command (\texttt{name}) which sets the options given by \texttt{(style)}. For instance, after issuing

\texttt{\todostyle{red}{\langle color=red, shadow\rangle}} the command \texttt{\todonote[red]{Stuff}} has the same effect as:

\texttt{\todonote[\langle color=red, shadow\rangle]{Stuff}}

Defined styles should be used as the first optional argument to \texttt{\todo}, asting they reset all predefined options to their defaults.

1.2 Package options

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

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

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.

Add 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, \ldots ) where the color of the inserted todonote marks the type of todo.

These options sets the default colors for the todo command. There are four colors that can be specified. The border color (default \texttt{bordercolor=black}) around
the inserted text, the color behind the inserted text (default backgroundcolor=orange),
the color of the inserted text (default textcolor=black) 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=length set the width of a todo item in the margin to length. The
width of inline todonotes defaults to the width of the current \linewidth but can
be adjusted using the inlinewidth option.

textsize=value 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 \tiny use textsize=tiny. The default value is textsize=normalsize.

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.

loadshadowlibrary If given, the shadows tikz library will be loaded. This is required for adding
shadows to the inserted todonotes.

shadow If the shadow option is given, the inserted todonotes will by default be displayed
with a gray shadow. For this to work, the package option loadshadowlibrary
must be provided. I expect that the option will trigger problems with tikz versions
prior to 2.0.

dvistyle When a document with todonotes is compiled with plain latex (to a dvi-file),
there is an issue with the visual appearance\footnote{The problem is placement of text inside the colored boxes.}. The option dvistyle changes the
appearance of the inserted todonotes to avoid this problem.

figwidth The figwidth=length option sets the default width of the figure inserted by
the \missingfigure command. The default value is \linewidth.

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}

\disable option can be given directly to the todo command. If given the
command has no effect.

color 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.
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}{\todo[color=green!40]{#1}}
An example that uses all of the color options is given below.
\todo[bgcolor=blue!20!white, bordercolor=red, textcolor=yellow]{Anything but default colors}.

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

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

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 ...}

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}

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}
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.

\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.

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:

\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:

\todo[fancyline]{Testing.}

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

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

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

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

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.

\todo[inline, inlinewidth=5cm, noinlinepar]{Testing the option inlinepar.}
1.4 Options for the \texttt{missingfigure} command

\texttt{figwidth} The \texttt{figwidth=\texttt{length}} option sets the width of the figure inserted by the \texttt{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.

\texttt{\begin{verbatim}
\texttt{missingfigure[figwidth=6cm]{Testing a long text string}}
\end{verbatim}}

Another usage of the option is when \texttt{missingfigure} is used in the \texttt{wrapfigure} environment.

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

\texttt{figheight} The \texttt{figheight=\texttt{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.

\texttt{\begin{verbatim}
\texttt{missingfigure[figheight=6cm]{Testing a long text string}}
\end{verbatim}}
The \texttt{figcolor=color} option sets the background color of inserted missing figures. The default color is \texttt{black!40}.

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

### 1.5 Options for the listoftodos command

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

\texttt{\listoftodos[I can be called anything]}

### 1.6 Known issues

#### 1.6.1 Package loading order

The todonotes package requires the following packages.

- \texttt{ifthen}
- \texttt{xkeyval}
- \texttt{xcolor}
- \texttt{tikz}
- \texttt{calc}
- \texttt{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’'.

10
It probably means that your LaTeX installation is outdated, as only newer versions of LaTeX driver for tikz supports the \texttt{remember picture} option. For additional info consult “Section 9.2.2 Producing PDF Output” in the tikz manual. 
\texttt{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 \texttt{marginparwidth} length. Try using the code below in your preamble to see if this cures the problem.

\texttt{\setlength{\marginparwidth}{2cm}}

If the todonotes are inserted in the wrong margin, the solution is the \texttt{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.

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

1.6.7 Todonotes in footnotes

Placing todonotes in footnotes throws a lot of errors. However inline notes seems to work fine, as shown here:\footnote{Note \todo[inline]{description}}

Richard Stanton comes with the following work around.

\texttt{\renewcommand{\marginpar}{\marginnote}}

1.6.8 Trouble with classicthesis.sty

Problem description on \texttt{tex.stackexchange.com}.

Solution by Stefan Kottwitz. The problem is caused by the redefinition of \texttt{marginpar} in classicthesis.sty. \texttt{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: \texttt{\let\marginpar\oldmarginpar}

\footnote{Note \texttt{description}}
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 \text command.

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

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

1.6.10 Conflict between marginpar and the standalone class

The standalone document class is not compatible with the \marginpar command, which is used in the todonotes package to insert contents in the side margins. A workaround is to replace the \marginpar with \marginnote. This is demonstrated here:

\documentclass[crop=false]{standalone}
\usepackage{todonotes}
\usepackage{marginnote}
\let\marginpar\marginnote
\begin{document}
Lorem ipsum
\todo{Bug}
\end{document}

1.6.11 Issues with user-defined styles

Using an option defined using \todostyle will reset all predefined options to their defaults. For instance, \todo[inline,newstyle]{Stuff} will not be inline; however \todo[newstyle,inline]{Stuff} will be.

\todostyle does not check if the name of the defined style is already one of the defined options, so, e.g., \todostyle{inline}{color=red} will redefine the inline option to produce red marginal notes.

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 todo\texttt{notes} 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 todo\texttt{notes} package can be loaded with the color\texttt{inlistoftodos} option, which adds small colored squares to the list of todos.

\usepackage[color\texttt{inlistoftodos}]{todo\texttt{notes}}
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.

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

The new command can now be used like shown below:

```latex
\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.

```latex
\todostyle{red}{color=red}
\todo[red,inline]{A red to do}
```

will produce a note like this:

A todo with user-defined style red

1.8.3 Enumerate todonotes

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

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

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

```latex
\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.

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

1: A numbered todonote.
2: Another numbered todonote.
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).

```latex
\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.

```latex
\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:

```latex
\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

```latex
\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.

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

It looks like seen here.

```latex
\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 \texttt{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={\texttt{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}
}

\begin{tcolorbox}
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}
\end{tcolorbox}

1.8.8 Link to list of todos

Using the \texttt{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[\%}
\newcommand{\todo}[1]{\todoTrans[\{\texttt{#1}\}]{}}

% Define a counter for the inserted todonotes.
\newcounter{todoListItems}
\newcommand{\todoTrans}[2][ ]{
% Increment counter
\addtocounter{todoListItems}{1}
\todo[\%}
\newcommand{\todo}[1]{\todoTrans[\{\texttt{#1}\}]{}}

% Define a counter for the inserted todonotes.
\newcounter{todoListItems}
\newcommand{\todoTrans}[2][ ]{
% Increment counter
\addtocounter{todoListItems}{1}
\todo[\%}
\newcommand{\todo}[1]{\todoTrans[\{\texttt{#1}\}]{}}
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.

```latex
\todoTrans{papirsflyver}
\todoTrans[inline]{damplokomotiv}
```

1.8.9 Numbered todonotes

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

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

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.}.
```

1.8.10 Combining several modifications

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

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

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]{\texthl{#1}\todo{#2}}
\fi
\makeatother
Example \hlfix{wrong text}{fix text} continues here.
```

1.8.14 Adding color coded levels of todos

Sometimes it can be beneficial to be able to distinguish between different types of todos. For this I would suggest the following approach:

```
\newcommand{\todoerror}[2]{\todo[color=red!70, #1]{#2}}
\newcommand{\todowarn}[2]{\todo[color=orange, #1]{#2}}
\newcommand{\todoremark}[2]{\todo[color=yellow!80!black, #1]{#2}}
\newcommand{\todohint}[2]{\todo[color=green!20, #1]{#2}}
\newcommand{\tododone}[2]{\todo[color=blue!50, #1]{#2}}
```

```
\todoerror[inline]{Testing}
\todoremark{Testing}
\todowarn[inline]{Testing}
\todohint{Testing}
```
1.8.15 Notes in both left and right margin

If you want to add many todonotes in your document, it can be beneficial to use both the left and right margins for the inserted notes. This can be achieved by defining a helper function the following way. The main drawback is that notes inserted using the `marginnote` command does not float and thus they can get to overlap. This idea was suggested by FranzAtGithub.

\usepackage{marginnote}
\usepackage{etoolbox}

\newtoggle{lmargin}
\newcommand{\alternatingtodo}[2][]{%\iftoggle{lmargin}{%\todo[#1]{#2}\togglefalse{lmargin}}{%\let\marginpar\marginnote\reversemarginpar\todo[#1]{#2}\toggletrue{lmargin}}}
2 Implementation

Identifies the package and loads the packages dependences.

\ProvidesPackage{todonotes}[2023/01/31]
\RequirePackage{ifthen}
\RequirePackage{xkeyval}
\RequirePackage{xcolor}
\RequirePackage{tikz}
\usetikzlibrary{positioning}
\RequirePackage{calc}

Implement a function for setting up the todonotes package.

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

Some default values are set

\newcommand{\@todonotes@todolistname}{Todo list}
\newcommand{\@todonotes@MissingFigureText}{Figure}
\newcommand{\@todonotes@MissingFigureUp}{Missing}
\newcommand{\@todonotes@MissingFigureDown}{figure}
\newcommand{\@todonotes@SetTodoListName}[1]{\renewcommand{\@todonotes@todolistname}{#1}}
\newcommand{\@todonotes@SetMissingFigureText}[1]{\renewcommand{\@todonotes@MissingFigureText}{#1}}
\newcommand{\@todonotes@SetMissingFigureUp}[1]{\renewcommand{\@todonotes@MissingFigureUp}{#1}}
\AtBeginDocument{
\ifx\undefined\phantomsection
\newcommand{\phantomsection}{}
\fi
\ifdim \marginparwidth < 2cm
\PackageWarning{todonotes}{The length marginparwidth is
less than 2cm and will most likely cause issues with the
appearance of inserted todonotes. The issue can be solved by adding a line like
\setlength{\marginparwidth}{2cm} prior to loading the todonotes package.}
\else\fi
}

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@MissingFigureText}{Figure}
\newcommand{\@todonotes@MissingFigureUp}{Missing}
\newcommand{\@todonotes@MissingFigureDown}{figure}
\newcommand{\@todonotes@SetTodoListName}[1]{\renewcommand{\@todonotes@todolistname}{#1}}
\newcommand{\@todonotes@SetMissingFigureText}[1]{\renewcommand{\@todonotes@MissingFigureText}{#1}}
\newcommand{\@todonotes@SetMissingFigureUp}[1]{\renewcommand{\@todonotes@MissingFigureUp}{#1}}
\newcommand{\@todonotes@SetMissingFigureDown}{\texttt{pendent}}
\newcommand{\@todonotes@SetMissingFigureUp}{\texttt{Figura}}
\newcommand{\@todonotes@SetMissingFigureText}{\texttt{Figura}}
\newcommand{\@todonotes@SetTodoListName}{Llista de feines pendent}
\@todonotes@reverseMissingFigureTrianglefalse
Create a counter, for storing the number of inserted todos.
\newcounter{@todonotes@numberoftodonotes}

Toggle whether the package should obey the global draft option.
\newif{\if@todonotes@obeyDraft}
\DeclareOptionX{obeyDraft}{\@todonotes@obeyDrafttrue}
\newif{\if@todonotes@isDraft}
\DeclareOptionX{draft}{\@todonotes@isDrafttrue}
\DeclareOptionX{draftcls}{\@todonotes@isDrafttrue}
\DeclareOptionX{draftclsnofoot}{\@todonotes@isDrafttrue}
\newif{\if@todonotes@obeyFinal}
\DeclareOptionX{obeyFinal}{\@todonotes@obeyFinaltrue}
\newif{\if@todonotes@isFinal}
\DeclareOptionX{final}{\@todonotes@isFinaltrue}
\DeclareOptionX{dvistyle}{\@todonotes@dviStyletrue}

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 your 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}{%
\begin{verbatim}
136  \{color\}
137     \renewcommand{\@todonotes@backgroundcolor}{#1}
138     \renewcommand{\@todonotes@linecolor}{#1}
139 \end{verbatim}

Make the background color of the notes as an option.

\begin{verbatim}
140 \define@key{todonotes.sty}{backgroundcolor}{\renewcommand{\@todonotes@backgroundcolor}{#1}}
141 \end{verbatim}

Make the text color of the notes as an option.

\begin{verbatim}
142 \define@key{todonotes.sty}{textcolor}{\renewcommand{\@todonotes@textcolor}{#1}}
143 \end{verbatim}

Make the line color of the notes as an option.

\begin{verbatim}
144 \define@key{todonotes.sty}{linecolor}{\renewcommand{\@todonotes@linecolor}{#1}}
145 \end{verbatim}

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

\begin{verbatim}
146 \newcommand{\@todonotes@defaulttickmarkheight}{0cm}
147 \define@key{todonotes.sty}{tickmarkheight}{\renewcommand{\@todonotes@defaulttickmarkheight}{#1}}
148 \end{verbatim}

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

\begin{verbatim}
150 \newif{\if@todonotes@prependcaptionglobal}
151 \@todonotes@prependcaptionglobalfalse
152 \DeclareOptionX{prependcaption}{\@todonotes@prependcaptionglobaltrue}
153 \end{verbatim}

Make the text width as an option.

\begin{verbatim}
154 \define@key{todonotes.sty}{textwidth}{\renewcommand{\@todonotes@textwidth}{#1}}
155 \end{verbatim}

Make the formatting of a note an option. Notes are formatted using \todoformat, which by default does nothing. If the option format=cmd is given, notes are formatted using \cmd instead. Default formatting can also be changed by redefining \todoformat.

\begin{verbatim}
155 \newcommand{\todoformat}{\{#1\}}
156 \define@key{todonotes.sty}{format}{\renewcommand{\todoformat}{\@nameuse{#1}}}
157 \end{verbatim}

Make the text size as an option, accept both size and textsize.

\begin{verbatim}
158 \newcommand{\todoformat}{\{\@nameuse{#1}\}}
159 \define@key{todonotes.sty}{format}{\renewcommand{\todoformat}{\@nameuse{#1}}}
160 \end{verbatim}

Make the text width as an option.

\begin{verbatim}
160 \define@key{todonotes.sty}{textwidth}{\renewcommand{\@todonotes@textwidth}{#1}}
161 \end{verbatim}

Add option for shadows behind the inserted notes

\begin{verbatim}
162 \newif{\if@todonotes@shadowlibraryloaded}
163 \@todonotes@shadowlibraryloadedfalse
164 \DeclareOptionX{loadshadowlibrary}{\usetikzlibrary{shadows}}
165 \\@todonotes@shadowlibraryloadedtrue
166 \newcommand{\@todonotes@shadowenabledbydefault}{noshadow}
167 \DeclareOptionX{shadow}{\renewcommand{\@todonotes@shadowenabledbydefault}{shadow}}
168 \DeclareOptionX{shadow}{\renewcommand{\@todonotes@shadowenabledbydefault}{\{\{\}}}
169 \end{verbatim}

23
Add option for the default width of the figure inserted with \texttt{missingfigure}.

\begin{macrocode}
\setkeys{todonotes}{figwidth}{#1}
\setkeys{todonotes}{figheight}{#1}
\setkeys{todonotes}{figcolor}{#1}
\end{macrocode}

Make the text width as an option.

\begin{macrocode}
% Finally process the given options.
% \begin{macrocode}
\ProcessOptionsX*
\if@todonotes@disabled
\else
\if@todonotes@obeyDraft
\@todonotes@disabledtrue
\if@todonotes@isDraft
\@todonotes@disabledfalse
\fi
\fi
\if@todonotes@obeyFinal
\@todonotes@disabledfalse
\if@todonotes@isFinal
\@todonotes@disabledtrue
\fi
\fi
\fi
\end{macrocode}

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.

\begin{macrocode}
gdef\@todonotes@currentlinecolor\@todonotes@linecolor\% 
gdef\@todonotes@currentbackgroundcolor\@todonotes@backgroundcolor\% 
gdef\@todonotes@currenttextcolor\@todonotes@textcolor\% 
gdef\@todonotes@currentbordercolor\@todonotes@bordercolor\% 
\setkeys{todonotes}{color}{#1}
\gdef\@todonotes@currentlinecolor{#1}\% 
\gdef\@todonotes@currentbackgroundcolor{#1}\% 
\gdef\@todonotes@currenttextcolor{#1}\% 
\gdef\@todonotes@currentbordercolor{#1}\% 
\end{macrocode}

Toggle whether there is a shadow behind the inserted notes.
Define height of the inserted tickmark.
\newcommand{\@todonotes@tickmarkheight}{#1}\%

Make formatting of notes a per-note option as well, by redefining the internal \@todonotes@format command, which is then used in formatting the actual note.
\newcommand{\@todonotes@format}{\todoformat}\%
\define@key{todonotes}{format}{\renewcommand{\@todonotes@format}{\@nameuse{#1}}}\%

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

Should the todo item be disabled?
\newif{\@todonotes@localdisable}\%
\define@key{todonotes}{disable}{\@todonotes@localdisabletrue}\%
\define@key{todonotes}{nodisable}{\@todonotes@localdisablefalse}\%

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

Should the todo item be displayed inline?
\newif{\@todonotes@inlinenote}\%
\define@key{todonotes}{inline}{\@todonotes@inlinenotetrue}\%
\define@key{todonotes}{noline}{\@todonotes@inlinenotefalse}\%
\newif{\@todonotes@prependcaption}\%
\define@key{todonotes}{prepend}{\@todonotes@prependcaptiontrue}\%
\define@key{todonotes}{noprepend}{\@todonotes@prependcaptionfalse}\%

Should the note in the margin be connected to the insertion point in the text? (does only work if line=true)
\newif{\@todonotes@fancyline}\%
\define@key{todonotes}{fancyline}{\@todonotes@fancylinetrue}\%
\define@key{todonotes}{nofancyline}{\@todonotes@fancylinetrue}\%

Author option.
\newcommand{\@todonotes@author}{}\%
\newif{\@todonotes@authorgiven}\%
\define@key{todonotes}{author}{\renewcommand{\@todonotes@author}{#1}\@todonotes@authorgiventrue}\%
\define@key{todonotes}{noauthor}{\@todonotes@authorgivenfalse}\%
Should the text in the list of todos be different from the text in the todonote?

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

Change the current figure width and height.
\newcommand{\@todonotes@currentfigwidth}{\@todonotes@figwidth}
define@key{todonotes}{figwidth}{\renewcommand{\@todonotes@currentfigwidth}{#1-2pt}}
\newcommand{\@todonotes@currentfigheight}{\@todonotes@figheight}
define@key{todonotes}{figheight}{\renewcommand{\@todonotes@currentfigheight}{#1-2pt}}
\newcommand{\@todonotes@currentfigcolor}{\@todonotes@figcolor}
define@key{todonotes}{figcolor}{\renewcommand{\@todonotes@currentfigcolor}{#1}}

Change the width of an inline note.
\newcommand{\@todonotes@inlinewidth}{\linewidth}
define@key{todonotes}{inlinewidth}{\renewcommand{\@todonotes@inlinewidth}{#1}}

Change if inline note is written to a separate line or not.
\newif{\@todonotes@inlinepar}
\@todonotes@inlinepartrue
define@key{todonotes}{inlinepar}{\@todonotes@inlinepartrue}
define@key{todonotes}{noinlinepar}{\@todonotes@inlineparfalse}

Preset values of the options
\presetkeys{todonotes}{linecolor=\@todonotes@linecolor,\
backgroundcolor=\@todonotes@backgroundcolor,\
textcolor=\@todonotes@textcolor,\
bordercolor=\@todonotes@bordercolor,\
format=todoformat,\
tickmarkheight=\@todonotes@defaulttickmarkheight,\
nofancyline,\
ndsiable,\
noinline,\
nocaption,\
noauthor,\
\@todonotes@shadowenabledbydefault,\
figwidth=\@todonotes@figwidth,\
figheight=\@todonotes@figheight,\
figcolor=\@todonotes@figcolor,\
line, list,\
inlinenewidth=\linewidth,\
inelinepar}{}
```

26
2.3 The main code part

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

```latex
\if@todonotes@disabled%
\newcommand{\listoftodos}{\@todo}
\newcommand{\missingfigure}{\@todo}
\else % \if@todonotes@disabled
Define the \listoftodos command and define the appearance of the list of todos.
\newcommand{\listoftodos}[1]{\@starttoc{tdo}}
\newcommand{\l@todo}{\@dottedtocline{1}{0em}{2.3em}}
Define styles used by the todo command
\tikzstyle{notestyleraw} = [draw=\@todonotes@currentbordercolor, fill=\@todonotes@currentbackgroundcolor, text=\@todonotes@currenttextcolor, line width=0.5pt, text width = \@todonotes@textwidth - 1.6 ex - 1pt, inner sep = 0.8 ex, rounded corners=4pt]
\@todo Define the \@todo command.
\newcommand{\@todo}[2]{% Use the global value for determining the default prepend behavior.
\if@todonotes@prependcaptionglobal%\@todonotes@prependcaptiontrue%\else%\@todonotes@prependcaptionfalse%\fi%
\renewcommand{\@todonotes@text}{#2}%\renewcommand{\@todonotes@caption}{#2}%\setkeys{todonotes}{#1}%
Add shadows to the inserted todonotes.
\if@todonotes@useshadow%\if@todonotes@shadowlibraryloaded%
\tikzstyle{notestyle} = [notestyleraw, general shadow={shadow xshift=0.5ex, shadow yshift=-0.5ex, opacity=1,fill=black!50}]%\else%
\PackageWarning{todonotes}{Trying to put a shadow below a todonote, but the loadshadowlibrary option was not given when loading the todonotes package}%\fi%
\else%
\PackageWarning{todonotes}{Trying to put a shadow below a todonote, but the loadshadowlibrary option was not given when loading the todonotes package}%\fi%}
```

\l@todo
Define the \l@todo command.
\newcommand{\l@todo}{\@dottedtocline{1}{0em}{2.3em}}
Define styles used by the todo command
\tikzstyle{notestyleraw} = [draw=\@todonotes@currentbordercolor, fill=\@todonotes@currentbackgroundcolor, text=\@todonotes@currenttextcolor, line width=0.5pt, text width = \@todonotes@textwidth - 1.6 ex - 1pt, inner sep = 0.8 ex, rounded corners=4pt]
\@todo Define the \@todo command.
\newcommand{\@todo}[2]{% Use the global value for determining the default prepend behavior.
\if@todonotes@prependcaptionglobal%\@todonotes@prependcaptiontrue%\else%\@todonotes@prependcaptionfalse%\fi%
\renewcommand{\@todonotes@text}{#2}%\renewcommand{\@todonotes@caption}{#2}%\setkeys{todonotes}{#1}%
Add shadows to the inserted todonotes.
\if@todonotes@useshadow%\if@todonotes@shadowlibraryloaded%
\tikzstyle{notestyle} = [notestyleraw, general shadow={shadow xshift=0.5ex, shadow yshift=-0.5ex, opacity=1,fill=black!50}]%\else%
\PackageWarning{todonotes}{Trying to put a shadow below a todonote, but the loadshadowlibrary option was not given when loading the todonotes package}%\fi%
\else%
\PackageWarning{todonotes}{Trying to put a shadow below a todonote, but the loadshadowlibrary option was not given when loading the todonotes package}%\fi%}
```
Update notestyles

\tikzstyle{notestyleleft} = [notestyle, left]
\tikzstyle{connectstyle} = [thick, draw=\@todonotes@currentlinecolor]
\tikzstyle{inlinenotestyle} = [notestyle, text width=\@todonotes@inlinewidth - 1.6 ex - 1 pt]

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

\if@todonotes@localdisable
\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.
\addtocounter{@todonotes@numberoftodonotes}{1}
\if@todonotes@appendtolistoftodos
  \phantomsection
  \if@todonotes@captiongiven
    \else
      \renewcommand{\@todonotes@caption}{#2}
    \fi
    \@todonotes@addElementToListOfTodos
  \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
  \@todonotes@drawInlineNote
\else
  \@todonotes@drawMarginNoteWithLine
\fi
\if@todonotes@localdisable
\Define helper function drawMarginNoteWithLine.
\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.
\vspace*{-\parskip} % backup if we are already in vertical mode
\vspace{-\baselineskip} % (and don’t loose that space after a pagebreak ...
In the book document class (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.

addElementToListOfTodos Define helper function addElementToListOfTodos.

\newcommand{\@todonotes@addElementToListOfTodos}{%  
  \if@todonotes@colorinlistoftodos%  
    \addcontentsline{tdo}{todo}{%  
      \fcolorbox{\@todonotes@currentbordercolor}{\@todonotes@currentbackgroundcolor}{\textcolor{\@todonotes@currentbackgroundcolor}{o}}%  
        \@todonotes@caption}%  
  \else%  
    \addcontentsline{tdo}{todo}{\@todonotes@caption}%  
  \fi}%

drawInlineNote Define helper function useSizeCommand.
\newcommand{\@todonotes@useSizeCommand}{%  
  \ifcsname \expandafter\string\@todonotes@sizecommand\endcsname%  
    \csname \expandafter\string\@todonotes@sizecommand\endcsname%  
  \else%  
    \@todonotes@sizecommand%  
  \fi%}

drawInlineNote Define helper function drawInlineNote.
\newcommand{\@todonotes@drawInlineNote}{%  
  \if@todonotes@dviStyle%  
    \begin{tikzpicture}%  
      \draw node[\@todonotes@inlinenotestyle] {};%  
    \end{tikzpicture}%  
  \if@todonotes@inlinepar\par\noindent\fi%  
    \@todonotes@useSizeCommand%  
  \if@todonotes@authorgiven%  
    \@todonotes@author:\@todonotes@format{\@todonotes@text}%  
  \else%  
    \@todonotes@useSizeCommand\@todonotes@format{\@todonotes@text}%  
  \fi%  
  \if@todonotes@inlinepar\par\noindent\fi%  
\}
\begin{tikzpicture}[remember picture]
\draw node[inlinenotestyle] {};
\end{tikzpicture}
\if@todonotes@inlinepar\par\fi\
\else\
{\if@todonotes@inlinepar\par\noindent\fi
\begin{tikzpicture}[remember picture]
\draw node[inlinenotestyle,font=\@todonotes@useSizeCommand]{
\if@todonotes@authorgiven
\noindent \@todonotes@author:\,\@todonotes@format{\@todonotes@text}
\else
\@todonotes@useSizeCommand\@todonotes@format{\@todonotes@text}
\fi};
\end{tikzpicture}\par
\fi}\
\fi}
\end{tikzpicture}

\begin{tikzpicture}
\node (inNote) {\vphantom{\@todonotes@useSizeCommand X}};
\if@todonotes@authorgiven
\draw node[notestyle,font=\@todonotes@useSizeCommand,anchor=north] (inNote) at (X.north)
{\@todonotes@author};
\node(Y)\[below=of X]\[,;\%
\draw node[notestyle,font=\@todonotes@useSizeCommand,anchor=north] (inNote) at (X.south)
{\@todonotes@format{\@todonotes@text}};
\else
\draw node[notestyle,font=\@todonotes@useSizeCommand,anchor=north] (inNote) at (X.north)
{\@todonotes@format{\@todonotes@text}};
\fi\end{tikzpicture}\hbadness \originalHbadness
\else
\vphantom{\@todonotes@useSizeCommand X}};
\if@todonotes@authorgiven
\draw node[notestyle,font=\@todonotes@useSizeCommand,anchor=north] (inNote) at (X.north)
{\@todonotes@author};
\node(Y)\[below=of X]\[,;\%
\draw node[notestyle,font=\@todonotes@useSizeCommand,anchor=north] (inNote) at (X.south)
{\@todonotes@format{\@todonotes@text}};
\else
\draw node[notestyle,font=\@todonotes@useSizeCommand,anchor=north] (inNote) at (X.north)
{\@todonotes@format{\@todonotes@text}};
\fi\end{tikzpicture}
drawLineToRightMargin Define helper function drawLineToRightMargin.

```latex
\newcommand{\@todonotes@drawLineToRightMargin}{%}
\if@todonotes@line%
\if@todonotes@fancyline%
\tikz[remember picture, overlay]{%}
\tikzstyle{both}=[line width=3pt, draw, opacity=0.15]%
\tikzstyle{line}=[shorten >=5pt, line cap=round]%
\tikzstyle{head}=[shorten >=-1pt, dash pattern=on 0pt off 1pt, ->]%
\foreach \s in {line,head}{%
\draw[both,\s]%(inNote.north west).. controls +(0:0) and +(90:1.5)..([yshift=1ex] inText);%
}%
\else%
\begin{tikzpicture}[remember picture, overlay]%
\draw[connectstyle]%(inNote.north west) -- ([yshift=-0.2cm + \@todonotes@tickmarkheight] inText)%
\end{tikzpicture}%
\fi%
\fi}%
\draw[both,\s]%(inNote.north west) -- ([yshift=-0.2cm] inText) -- ([yshift=-0.2cm] inNote.west) -- (inNote.west);%
\missingfigure
Defines the \missingfigure macro.
```

```latex
\newcommand{\@todonotes@MissingFigureText}{%}
\if@todonotes@line%
\if@todonotes@fancyline%
\tikz[remember picture, overlay]{%}
\tikzstyle{both}=[line width=3pt, draw, opacity=0.15]%
\tikzstyle{line}=[shorten >=5pt, line cap=round]%
\tikzstyle{head}=[shorten >=-1pt, dash pattern=on 0pt off 1pt, ->]%
\foreach \s in {line,head}{%
\draw[both,\s]%(inNote.north east) -- ([yshift=0:0] inText) -- ([yshift=0:1.5] inNote.east) -- (inNote.east);%
}%
\else%
\begin{tikzpicture}[remember picture, overlay]%
\draw[connectstyle]%(inNote.north east) -- ([yshift=-0.2cm + \@todonotes@tickmarkheight] inText)%
\end{tikzpicture}%
\fi%
\fi}%
\missingfigure[2][]{%}
\setkeys{todonotes}{#1}%
\addcontentsline{tdo}{todo}{\@todonotes@MissingFigureText: #2}%
\par
\noindent
```
510 \hfill
511 \begin{tikzpicture}
512 \draw[fill=\@todonotes@currentfigcolor, draw = black!40, line width=2pt]
513 (-2, -0.5*\@todonotes@currentfigheight-0.5cm)
514 rectangle +(\@todonotes@currentfigwidth, \@todonotes@currentfigheight);
515 \draw (2, -0.5) node[right, text
516 width=\@todonotes@currentfigwidth-4.5cm, font=\@todonotes@useSizeCommand] {#2};
517 \draw[red, fill=white, rounded corners = 5pt, line width=10pt]
518 (30:2cm) -- (150:2cm) -- (270:2cm) -- cycle;
519 \draw (0, 0.3) node {\@todonotes@MissingFigureUp};
520 \draw (0, -0.3) node {\@todonotes@MissingFigureDown};
521 \end{tikzpicture}\hfill
522 \null\par
523 }% Ending \missingfigure command
524 \fi% Ending \@todonotes@ifdisabled
525 \fi\% Ending \@todonotes@ifdisabled
526 \if@todonotes@disabled
527 \else
528 \addcontentsline{toc}{\@ifundefined{chapter}{section}{chapter}}{\@todonotes@todolistname}\fi
529 \fi
530 \fi
531 }
532 \newcommand{\todo}[2][]{%\if@todonotes@inlinepar
533 \% Needed to output any dangling \item of a noskip section (see #36):
534 \if@inlabel \leavevmode \fi
535 \if@noskipsec \leavevmode \fi
536 \if@todonotes@inlinepar
537 \ifhmode
538 \@bsphack
539 \@todonotes@vmodetrue
540 \else
541 \@savsf\@m
542 \@savsk\z@
543 \@todonotes@vmodefalse
544 \fi
545 {\@todo[#1][#2]}%
546 \@esphack%
547 \if@todonotes@vmode \par \fi
548 \else%
549 \@todo[#1][#2]%
550 \fi
551 \newif\if@todonotes@vmode
552 \newcommand{\todostyle}{\@ifundefined{name}{\@ifundefined{style}{\@todo}{}}}%
553 Define the \todo command as a redirection to \@todo.
554 \newcommand{\todo}[2][]{%
555 \% Needed to output any dangling \item of a noskip section (see #36):
556 \if@inlabel \leavevmode \fi
557 \if@noskipsec \leavevmode \fi
558 \if@todonotes@inlinepar
559 \ifhmode
560 \@bsphack
561 \@todonotes@vmodetrue
562 \else
563 \@savsf\@m
564 \@savsk\z@
565 \@todonotes@vmodefalse
566 \fi
567 {\@todo[#1][#2]}%
568 \@esphack%
569 \if@todonotes@vmode \par \fi
570 \else%
571 \@todo[#1][#2]%
572 \fi
573 \newif\if@todonotes@vmode
574 \newcommand{\todostyle}{\@ifundefined{name}{\@ifundefined{style}{\@todo}{}}}%
575 Define the \todostyle macro. \ todostyle\{<name>\}\{<style>\} defines an xkeyval option \{<name>\} that, when called in a \todo or \missingfigure command, sets the preset options followed by \{<style>\}.
\define@key{todonotes}{#1}{%}
\setkeys{todonotes}{#2}}
Change History

0.1
General: The first version of the package .......................... 1

0.2
General: Updated the option handling 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

0.2.2
General: Added a missingfigure command ...................... 1

0.2.3
General: Made a dependency on the calc package .............. 1

0.3
General: Delayed the requirements for the hyperref package until begin document and added an optional argument to the todo command for adding inline todos (Idea from Patrick Toche) .............. 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

0.4
General: Modified the behaviour of the inline todos, to avoid empty lines around the inline todos. .......................... 1

0.4.1
General: Added the option colorinlistoftodos which inserts a small box with the used fillcolor of the todos in the list of todos. .............. 1

0.4.2
General: Fixed a bug with the disable option to the package. 1

0.5
General: Created a dtx file containing both source code and documentation of the package .......................... 1

0.5.1
General: Updated the documentation .......................... 1

0.5.2
General: Fixed a bug that prevented the usage of the option french for babel. Bug report by Thomas Braun. .... 1

0.6
General: Added the caption option to the todo command. .... 1

0.6.1
General: Added a new usecase with decreased line spacing. ... 1

0.6.2
General: Added a usecase by Fabrice Niessen. .............. 1

0.7
General: Added language options on request from Peter Zimmermann. .............. 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. 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

34
0.7.6
General: Added a textsize option for the package and the prepend / noprepend option for the todo command. 

0.8
General: Added three new translations: french, spanish, and catalan thanks to Richard Dominique and Joan Queralt. Improved the visual appearence 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. 

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. 

0.8.2
General: Italian translation by Gustavo Cevolani. Removed the dependence on the hyperref package. 

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. 

0.8.4
General: Fixed a bug related to the obeyDraft option. 

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. 

0.8.6
General: Added a portuguese translation by Og DeSouza. 

0.8.7
General: Updated portuguese translation. Added a german alias for the german translation suggested by Michael Niedermair. 

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. 

0.8.9
General: Added a dutch translation by Ruben Vermeersch. 

0.9.0
General: Added a english option as suggested by Marco Berghoff. 

0.9.1
General: Added the todototoc command by idea from Sven Augustin. 

0.9.2
General: Use chapter (if available) for the list of todos heading. 

0.9.3
General: Make an internal definition of the todo command, for easing redefinition of the command behaviour. 

0.9.4
General: Make the disable option work on a local scale. 

0.9.5
General: Code simplification by extracting functionality to smaller macros. 

0.9.6
General: Give fontsize to TikZ. Align notes with line where note is set. Added new option fancyline. Patches by Benjamin Kellermann. 

0.9.7
General: Updated documentation. 

0.9.8
General: Suppress warnings about underfull / overful boxes generated by the inserted todonotes. Patch by Peter M Schuler.
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

1.0.0
General: Mention trouble with the classthesis 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

1.0.1
General: Fix spacing issues reported by Jonathan Zachhuber and Brent Longborough. Added figheight option to the missingfigure command as suggested by Kim Albertsson. ................. 1

1.0.2
General: Added Swedish translation by Emil Lundberg. Added usecase by Tobias Winchen. Mentioned that default arguments can be set 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

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. ....... 1

1.0.4
General: Restructured documentation and placed some examples in the doc/examples subdirectory. ...... 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. ................. 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

1.0.7
General: Changed the warning about marginparwidth to a proper warning message. ...... 1

1.0.8
General: Improved spacing 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

1.1.0
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 customizable height. Steamline which options that are accepted to size commands (both commands (backslash tiny) and text (tiny) are now supported). ................. 1

1.1.1
General: Implemented shadow and noshadow options for the todo command. ................. 1
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1.2</td>
<td>General: Fix issue 36 and 37. todos with shadows. Actually use the tickmarkheight option as suggested by José Francisco Loff library by default.</td>
</tr>
<tr>
<td>1.1.3</td>
<td>General: Fix issue 48 and make colors used in the last todo globally available. Described a workaround for using the standalone document class.</td>
</tr>
<tr>
<td>1.1.4</td>
<td>General: Fix issue 51 by avoid loading the tikz shadows library by default.</td>
</tr>
<tr>
<td>1.1.5</td>
<td>General: Fix issue 54 related to additional spacing around</td>
</tr>
<tr>
<td>1.1.6</td>
<td>General: Fix issue 64 by adding the command definitodestyle and the format option to the todo and missingfigure commands. Pull request by rzach.</td>
</tr>
</tbody>
</table>