zebra-goodies — Easy Notes Taking[*]

Ruini Xue[†]

Released 2026/03/21

Abstract

The zebra-goodies package provides a lightweight set of macros for inline note-taking during document writing. It is designed to be simple and practical for both solo and collaborative workflows. Five built-in commands—`\todo`, `\comment`, `\note`, `\fixed`, and `\placeholder`—cover common use cases out of the box, and `\zebranewnote` lets you define additional note types as needed. Notes are automatically numbered per type, marked with a `\dbend` symbol in the nearest margin, and summarised with a summary table plus a detailed note list at the end of the document. Passing the `final` option suppresses all notes for production output.

Contents

[*] This file describes version v1.1.0, last revised 2026/03/21.

[†] E-mail: xueruini@gmail.com

1 Introduction

Many note-taking and to-do packages exist for LaTeX, but most fall into one of two traps: they either offer an overwhelming feature set that tries to cover every conceivable use case, or they clutter the margins with oversized colourful boxes and arrows that make the document hard to read.

zebra-goodies takes a different approach. It aims to be *simple* —intuitive commands with only the arguments you actually need—and *good enough*—notes appear inline with a small visual cue in the margin, keeping the document readable while still making annotations easy to spot. Each note type is automatically numbered, and a summary table plus a detailed note list at the end of the document serve as a gentle reminder to address them before the final version.

The package requires LaTeX 2022-06-01 or later.

2 Installation

The package is supplied in `dtx` format and as a pre-extracted zip file. The latter is the most convenient option for most users: simply unzip it into your local texmf directory and run `texhash` to update the file-name database, or unzip the files directly into your working directory. To unpack the `dtx` yourself, run `tex zebra-goodies.dtx` to extract the package, or `latex zebra-goodies.dtx` to extract it and typeset the documentation at the same time.[1]

3 Using the package

Load the package in the preamble with any desired options.

```
\usepackage[<options>]{zebra-goodies}
```

3.1 Package Options

`draft`
`final` These two options are complementary. By default `draft` is active: all notes are typeset inline and a summary table plus a detailed note list are appended at the end of the document. Setting `final` (or `draft=false`) suppresses all notes and the generated lists, producing clean output ready for distribution.

`microtypeexpansion` Controls whether microtype is loaded with font expansion enabled. Default: `true`. Set `microtypeexpansion=false` or use the complementary option `nomicrotypeexpansion` to disable expansion while keeping microtype loaded. zebra-goodies ensures microtype is loaded. Engines without font expansion support still fall back to `expansion=false` automatically.

`hyperref` Controls whether the hyperref package is loaded by zebra-goodies. Default: `true`. Set `hyperref=false` or use the complementary option `nohyperref` to disable automatic loading.

`listofnotes` Controls whether the summary table and detailed note list are printed at the end of the document. Default: `true`. Set `listofnotes=false` or use the complementary option `nolistofnotes` to suppress them.

[1] Running `latexmk zebra-goodies.dtx` is even more convenient as it handles multiple compilation passes automatically.

`symbol` Sets the margin symbol used by all note types. Default: `\textdbend` (from manfnt). For example, `symbol=\textdagger` replaces the dangerous-bend sign with a dagger. Individual note types can override this via the optional argument of `\zebranewnote`.

3.2 Notes Macros

`\todo` `\todo[⟨name⟩]{⟨note content⟩}`

The primary command provided by zebra-goodies is `\todo`. It inserts an inline note in the current paragraph, typeset in a predefined colour and marked with a `\dbend` symbol in the nearest margin. The mandatory ⟨*note content*⟩ describes the task; the optional ⟨*name*⟩ specifies who is responsible for addressing it, which is particularly useful during collaborative writing.

```
A very simple example of a todo note \todo{how should we go to the lake?},
and the result will be presented right below the code snippet. Please
compare the code and the result.
```

1 A very simple example of a todo note [Todo 1: how should we go to the lake?], and the result will be presented right below the code snippet. Please compare the code and the result.

The following example shows how to assign notes to specific people. Notes of the same type are numbered sequentially, and assignees are prefixed with `@`.

```
Another one with an assignee is like \todo[tom]{you should buy the ticket},
and you will find the same type of notes are numbered. And the assignees are
being \texttt{@}-ed right before the text. And another one with many
assignees \todo[lucy, jessie, tom]{prepare the foo}.
```

2 Another one with an assignee is like [Todo 2`@tom`: you should buy the ticket], and you will find the same type of notes are numbered. And the assignees are being `@`-ed right before the text.
3 And another one with many assignees [Todo 3`@lucy, jessie, tom`: prepare the foo].

Because notes are set as inline text, they work inside moving arguments such as `\section` and `\caption` without issues.

```
 \section{Introduction\todo[jerry]{please figure out a better name}}
 \begin{figure}
   \centering
   \caption{Speed vs distance. \todo{need to insert the figure}}
 \end{figure}
```

`\note` `\note[⟨name⟩]{⟨note text⟩}`

`\comment` `\comment[⟨name⟩]{⟨comment text⟩}`

`\fixed` `\fixed[⟨name⟩]{⟨fixed text⟩}`

`\placeholder` `\placeholder[⟨name⟩]{⟨placeholder text⟩}`

These commands share the same syntax and behaviour as `\todo`; they differ only in name and colour, providing semantic distinction for different annotation purposes.

If any of these short names conflicts with another loaded package, zebra-goodies will *not* overwrite the existing definition. Instead, it issues a warning and you should use the prefixed form (e.g. `\zebracomment` instead of `\comment`). The following example mixes several note types. Note that `\zebracomment` is used here because `\comment` is already defined by l3doc.

```
We may want to reorganise \note{how should we structure the intro?} this
part before the final submission. The experimental setup in Section~2 has
already been reviewed by a collaborator \zebracomment[tom]{the setup
description looks clear now}. Results are presented in the following tables
and figures, but some of them are still missing. The discussion section has
been revised \placeholder[lucy, tom]{good job!} and all previous issues
addressed \fixed[John]{updated the conclusion}. Finally, the list of
references needs a second pass before we can submit \todo{check bibliography
entries for formatting}. With these items resolved, \note[who]{anything
else?} the draft should be ready to go.
```

We may want to reorganise [Note 1: how should we structure the intro?] this part before
1 the final submission. The experimental setup in Section 2 has already been reviewed by a
collaborator [Comment 1@tom: the setup description looks clear now]. Results are presented
1 in the following tables and figures, but some of them are still missing. The discussion section
has been revised [Placeholder 1@lucy, tom: good job!] and all previous issues addressed
1 1 ✓ [Fixed 1@John: updated the conclusion]. Finally, the list of references needs a second pass before
we can submit [Todo 4: check bibliography entries for formatting]. With these items resolved,
4 [Note 2@who: anything else?] the draft should be ready to go.
2

`\zebratodo` `\zebratodo[⟨name⟩]{⟨note text⟩}`

Each built-in note command has a prefixed alias that is always available, regardless of name conflicts. If `\todo` is already taken by another package, use `\zebratodo` instead. The same applies to all other types.

`\zebranote` `\zebranote[⟨name⟩]{⟨note text⟩}`

`\zebracomment` `\zebracomment[⟨name⟩]{⟨comment text⟩}`

`\zebrafixed` `\zebrafixed[⟨name⟩]{⟨fixed text⟩}`

`\zebraplaceholder` `\zebraplaceholder[⟨name⟩]{⟨placeholder text⟩}`

\zebranewnote

\zebranewnote{⟨*note name*⟩}{⟨*xcolor name*⟩}[⟨*symbol*⟩]

Creates a new note type. The ⟨*note name*⟩ becomes the command name (e.g. passing `question` creates `\question` and `\zebraquestion`), and ⟨*xcolor name*⟩ sets its colour. The colour must be a named colour already known to xcolor; define it with `\definecolor` or `\colorlet` beforehand if needed. The optional ⟨*symbol*⟩ overrides the global margin symbol for this note type only. If omitted, the global `symbol` option is used.

```
\colorlet{mycyan}{cyan}
\zebranewnote{question}{mycyan}[\faQuestionCircle] % \usepackage{fontawesome}
When it moves to the next step, we should be fine.\question[who]{what's this?}
```

1? When it moves to the next step, we should be fine.[Question 1@who: what's this?]

\zebranotesymbol

\zebranotesymbol{⟨*note name*⟩}{⟨*symbol*⟩}

Changes the margin symbol of an existing note type. This is useful for overriding the symbol of built-in types such as `\fixed` in this document.

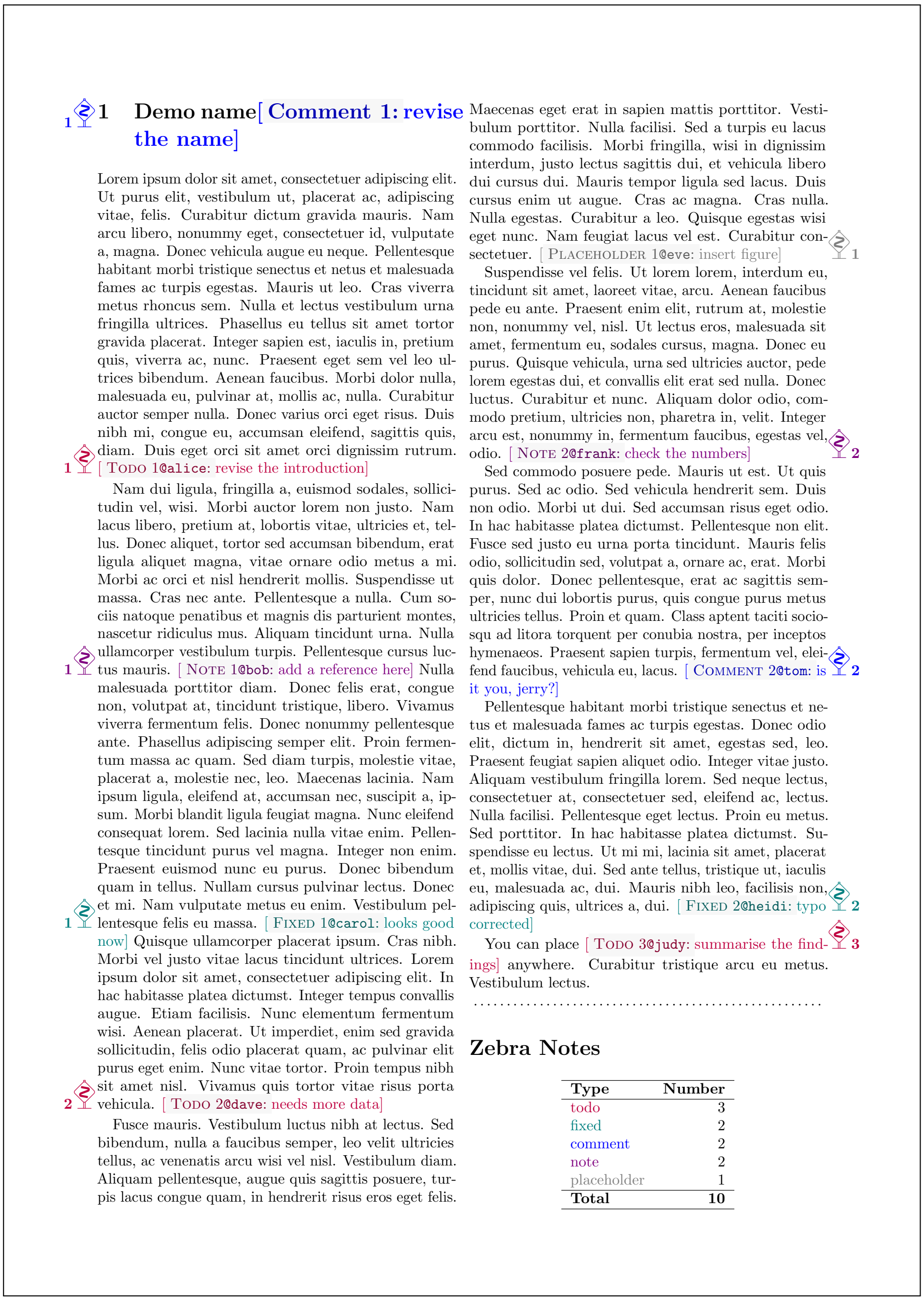
```
\zebranotesymbol{fixed}{$\surd$}
```

3.3 Two-column Support

In `twocolumn` documents, the `\dbend` symbol is automatically placed on the nearest margin: left margin for the left column, right margin for the right column. No special configuration is needed. This also works correctly in combination with the `twoside` option.

```
\documentclass[twocolumn,a4paper]{article}
\usepackage[margin=2cm]{geometry}
\usepackage{zebra-goodies}
\usepackage{lipsum}
\pagestyle{empty}
\begin{document}
\section{Demo name\comment{revise the name}}
\lipsum[1] \todo[alice]{revise the introduction}

\lipsum[2] \note[bob]{add a reference here}
\lipsum[3] \fixed[carol]{looks good now}
\lipsum[4] \todo[dave]{needs more data}

\lipsum[5] \placeholder[eve]{insert figure}

\lipsum[6] \note[frank]{check the numbers}

\lipsum[7] \comment[tom]{is it you, jerry?}

\lipsum[8] \fixed[heidi]{typo corrected}

You can place \todo[judy]{summarise the findings} anywhere.
\lipsum[9][10-11]
\end{document}
```

The code above produces the following output:



1 Demo name[Comment 1: revise the name]

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum. [Todo 1@alice: revise the introduction]

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris. [Note 1@bob: add a reference here] Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa. [Fixed 1@carol: looks good now] Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. [Todo 2@dave: needs more data]

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer. [Placeholder 1@eve: insert figure]

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio. [Note 2@frank: check the numbers]

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus. [Comment 2@tom: is it you, jerry?]

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec odio elit, dictum in, hendrerit sit amet, egestas sed, leo. Praesent feugiat sapien aliquet odio. Integer vitae justo. Aliquam vestibulum fringilla lorem. Sed neque lectus, consectetuer at, consectetuer sed, eleifend ac, lectus. Nulla facilisi. Pellentesque eget lectus. Proin eu metus. Sed porttitor. In hac habitasse platea dictumst. Suspendisse eu lectus. Ut mi mi, lacinia sit amet, placerat et, mollis vitae, dui. Sed ante tellus, tristique ut, iaculis eu, malesuada ac, dui. Mauris nibh leo, facilisis non, adipiscing quis, ultrices a, dui. [Fixed 2@heidi: typo corrected]

You can place [Todo 3@judy: summarise the findings] anywhere. Curabitur tristique arcu eu metus. Vestibulum lectus.

....

Zebra Notes

Type	Number
todo	3
fixed	2
comment	2
note	2
placeholder	1
Total	**10**

4 Implementation

```
⟨*package⟩
⟨@@=zebra⟩
```

Version data to start with.

```
\ProvidesExplPackage{zebra-goodies}
```

```
  {2026/03/21}
  {1.1.0}
  {Easy Notes Taking}
```

4.1 Package options

Five package options `draft`, `microtypeexpansion`, `hyperref`, `listofnotes`, and `symbol` are created using the kernel key–value interface available since LaTeX 2022-06-01.

```
\bool_new:N \l__zebra_draft_bool
\bool_new:N \l__zebra_microtypeexpansion_bool
\bool_new:N \l__zebra_hyperref_bool
\bool_new:N \l__zebra_listofnotes_bool
\seq_new:N \g__zebra_note_types_seq
\prop_new:N \g__zebra_note_colors_prop
\prop_new:N \g__zebra_note_public_alias_prop
\int_new:N \g__zebra_note_id_int
\tl_new:N \l__zebra_note_target_tl
\tl_new:N \l__zebra_summary_rows_tl
\int_new:N \l__zebra_total_notes_int
\prop_new:N \g__zebra_note_symbols_prop

\msg_new:nnn { zebra } { command-taken }
  {
    The~command~’\iow_char:N\\#1’~is~already~defined.~
    Use~’\iow_char:N\\zebra#1’~instead.
  }

\prg_new_conditional:Npnn \__zebra_if_package_loaded:n #1 { T , F , TF }
  {
    \cs_if_exist:cTF { ver@#1.sty }
      { \prg_return_true: }
      { \prg_return_false: }
  }

\keys_define:nn { zebra }
  {
    draft .bool_set:N = \l__zebra_draft_bool,
    draft .initial:n = true,
    final .meta:n = { draft = false },
    microtypeexpansion .bool_set:N = \l__zebra_microtypeexpansion_bool,
    microtypeexpansion .initial:n = true,
    nomicrotypeexpansion .meta:n = { microtypeexpansion = false },
    hyperref .bool_set:N = \l__zebra_hyperref_bool,
    hyperref .initial:n = true,
    nohyperref .meta:n = { hyperref = false },
    listofnotes .bool_set:N = \l__zebra_listofnotes_bool,
    listofnotes .initial:n = true,
    nolistofnotes .meta:n = { listofnotes = false },
    symbol .tl_set:N = \l__zebra_symbol_tl,
    symbol .initial:n = \textdbend,
  }
\ProcessKeyOptions [ zebra ]
\sys_if_engine_xetex:T
  { \bool_set_false:N \l__zebra_microtypeexpansion_bool }

```

```
\cs_new_protected:Npn \__zebra_setup_microtype:
  {
    \__zebra_if_package_loaded:nTF { microtype }
      {
        \bool_if:NF \l__zebra_microtypeexpansion_bool
          { \microtypesetup { expansion = false } }
      }
      {
        \bool_if:NTF \l__zebra_microtypeexpansion_bool
          { \RequirePackage{microtype} }
          { \RequirePackage[expansion=false]{microtype} }
      }
  }
\__zebra_setup_microtype:
\__zebra_if_package_loaded:nF { hyperref }
  {
    \bool_if:NT \l__zebra_hyperref_bool
      {
        \RequirePackage{hyperref}
        \hypersetup { pdfborder = { 0 0 0 } }
      }
  }
\RequirePackage{xcolor}
\RequirePackage{marginnote}
\cs_if_exist:NTF \dbend
  {
    \cs_set_eq:NN \__zebra_saved_dbend: \dbend
    \cs_undefine:N \dbend
    \RequirePackage{manfnt}
    \cs_set_eq:NN \dbend \__zebra_saved_dbend:
  }
  { \RequirePackage{manfnt} }
\cs_new_protected:Npn \__zebra_pdfstring_note:
  { \@ifnextchar [ { \__zebra_pdfstring_note_opt:w } { \use_none:n } }
\cs_new_protected:Npn \__zebra_pdfstring_note_opt:w [#1] #2 { }
\cs_new_protected:Npn \__zebra_apply_pdfstring_defs:
  {
    \__zebra_if_package_loaded:nT { hyperref }
      {
        \pdfstringdefDisableCommands
          {
            \seq_map_inline:Nn \g__zebra_note_types_seq
              {
                \cs_set_eq:cN { zebra##1 } \__zebra_pdfstring_note:
                \prop_if_in:NnT \g__zebra_note_public_alias_prop { ##1 }
                  { \cs_set_eq:cN { ##1 } \__zebra_pdfstring_note: }
              }
          }
      }
  }
\hook_gput_code:nnn { begindocument } { zebra-goodies }
  { \__zebra_apply_pdfstring_defs: }
```

4.2 Main notes macros

Various helper macros are defined before reaching out to the \todo commands.

Place the margin note on the nearest margin. Takes two arguments: #1 for the left margin (number then symbol) and #2 for the right margin (symbol then number), so the symbol always sits closest to the text column. In twocolumn mode, \marginpar's optional argument selects the left-margin variant automatically. In single-column mode, \marginnote is used with the right-margin variant as default.

```
\cs_new_protected:Npn \__zebra_margin_note:nn #1#2
  {
    \legacy_if:nTF { @twocolumn }
      {
        \mode_if_inner:TF
          { \marginnote{#2} }
          {
            \marginpar
              [ { \makebox[\marginparwidth]{#1} } ]
              { \makebox[\marginparwidth]{#2} }
          }
      }
      { \marginnote[#1]{#2} }
  }
\cs_new:Npn \__zebra_target:nn #1#2
  {
    \__zebra_if_package_loaded:nTF { hyperref }
      { \hypertarget{#1}{#2} }
      {#2}
  }
\cs_new:Npn \__zebra_link:nn #1#2
  {
    \__zebra_if_package_loaded:nTF { hyperref }
      { \hyperlink{#1}{#2} }
      {#2}
  }
\cs_new:Npn \__zebra_pageref:n #1
  {
    \__zebra_if_package_loaded:nTF { hyperref }
      { \pageref* {#1} }
      { \pageref {#1} }
  }
\cs_new:Npn \__zebra_prepend:nn #1#2
  { \tl_if_blank:nTF {#2} {} {#1#2} }
\cs_new:Npn \__zebra_capitalize_type:n #1
  { \text_uppercase:n { \tl_head:n {#1} } \tl_tail:n {#1} }
\cs_new:Npn \__zebra_note_count:n #1
  { \int_use:c { g__zebra_note_count_#1_int } }
\cs_new:Npn \__zebra_note_color:n #1
  { \prop_item:Nn \g__zebra_note_colors_prop {#1} }
\cs_new:Npn \__zebra_note_symbol:n #1
  {
    \prop_if_in:NnTF \g__zebra_note_symbols_prop {#1}
      { \prop_item:Nn \g__zebra_note_symbols_prop {#1} }
      { \l__zebra_symbol_tl }
  }
```

```
\cs_new_protected:Npn \__zebra_new_listbody:n #1
  { \tl_new:c { g__zebra_listbody_#1_tl } }
\cs_new:Npn \__zebra_use_listbody:n #1
  { \tl_use:c { g__zebra_listbody_#1_tl } }
\cs_new_protected:Npn \__zebra_record_note:nnnnnn #1#2#3#4#5#6
  {
    \tl_gput_right:cx { g__zebra_listbody_#1_tl }
      {
        \exp_not:N \__zebra_list_entry:nnnnnn
        { \exp_not:n {#1} }
        { \exp_not:n {#2} }
        {#3}
        { \exp_not:n {#4} }
        { \exp_not:n {#5} }
        {#6}
      }
  }
\cs_new_protected:Npn \__zebra_note:nnnn #1#2#3#4
  {
    \bool_if:NT \l__zebra_draft_bool
      {
      \int_gincr:c { g__zebra_note_count_#1_int }
      \int_gincr:N \g__zebra_note_id_int
      \tl_set:Nx \l__zebra_note_target_tl
        { zebranote.\int_use:N \g__zebra_note_id_int }
      \__zebra_record_note:nnnnnn
        {#1}
        {#2}
        { \__zebra_note_count:n {#1} }
        {#3}
        {#4}
        { \l__zebra_note_target_tl }
      \__zebra_target:nn { \l__zebra_note_target_tl } {}
      \exp_args:NV \label \l__zebra_note_target_tl
      \__zebra_margin_note:nn
        {\textcolor{#2}{%
          {\bfseries\__zebra_note_count:n {#1}}\kern1pt\__zebra_note_symbol:n {#1}}}
        {\textcolor{#2}{%
          \__zebra_note_symbol:n {#1}\kern1pt{\bfseries\__zebra_note_count:n {#1}}}}%
      \textcolor{#2}{[\colorbox[gray]{0.97}{%
          \textcolor{#2!70!black}{%
            \textsc{\MakeLowercase{\MakeUppercase#1}}~%
            \__zebra_note_count:n {#1}\texttt{\__zebra_prepend:nn {@}{#3}}:}} #4]}%
      }
  }
\cs_new_protected:Npn \__zebra_new_note_type:nn #1#2
  { \__zebra_new_note_type:nnn {#1} {#2} {} }
\cs_new_protected:Npn \__zebra_new_note_type:nnn #1#2#3
  {
    \seq_gput_right:Nn \g__zebra_note_types_seq {#1}
    \prop_gput:Nnn \g__zebra_note_colors_prop {#1} {#2}
    \tl_if_blank:nF {#3}
      { \prop_gput:Nnn \g__zebra_note_symbols_prop {#1} {#3} }
    \int_new:c { g__zebra_note_count_#1_int }
    \__zebra_new_listbody:n {#1}
    \exp_args:Nc \NewDocumentCommand { zebra#1 } { O{} m }
      { \__zebra_note:nnnn {#1}{#2}{##1}{##2} }
    \__zebra_if_package_loaded:nT { hyperref }
      { \__zebra_apply_pdfstring_defs: }
    \cs_if_exist:cTF {#1}
      { \msg_warning:nnn { zebra } { command-taken } {#1} }
      {
        \cs_set_eq:cc {#1} {zebra#1}
        \prop_gput:Nnn \g__zebra_note_public_alias_prop {#1} { true }
      }
  }
\cs_new_protected:Npn \__zebra_list_entry:nnnnnn #1#2#3#4#5#6
  {
    \par\noindent
    \textcolor{#2}{%
      \textbf{\__zebra_capitalize_type:n {#1}~#3}%
      \tl_if_blank:nF {#4} { \enspace \texttt{\__zebra_prepend:nn {@}{#4}} }}%
    \nobreak\dotfill
    \__zebra_link:nn {#6} { \__zebra_pageref:n {#6} }%
    \par
    \begingroup
      \leftskip=1.5em
      \parindent=0pt
      #5\par
    \endgroup
  }
\cs_new_protected:Npn \__zebra_print_note_group:n #1
  {
    \int_compare:nNnT { \__zebra_note_count:n {#1} } > { 0 }
      {
        \par\medskip
        \__zebra_use_listbody:n {#1}
      }
  }
\cs_new_protected:Npn \__zebra_summary_row:n #1
  {
    \int_compare:nNnT { \__zebra_note_count:n {#1} } > { 0 }
      {
        \int_add:Nn \l__zebra_total_notes_int { \__zebra_note_count:n {#1} }
        \tl_put_right:Nx \l__zebra_summary_rows_tl
          {
            \exp_not:N \textcolor
              { \__zebra_note_color:n {#1} }
              {#1}
            \exp_not:N &
            \__zebra_note_count:n {#1}
            \exp_not:N \\
          }
      }
  }
\cs_new_protected:Npn \__zebra_print_notes:
  {
    \tl_clear:N \l__zebra_summary_rows_tl
    \int_zero:N \l__zebra_total_notes_int
    \seq_map_inline:Nn \g__zebra_note_types_seq
      { \__zebra_summary_row:n {##1} }
    \tl_if_empty:NF \l__zebra_summary_rows_tl
      {
        \par\nobreak
        \noindent\dotfill\par
        \nobreak
        \section*{Zebra~Notes}
        \par \medskip
        \begin{center}
          \begin{tabular}{lr}
            \hline
            \textbf{Type} & \textbf{Number} \\\hline
            \tl_use:N \l__zebra_summary_rows_tl
            \hline
            \textbf{Total} & \textbf{\int_use:N \l__zebra_total_notes_int} \\
            \hline
          \end{tabular}
        \end{center}
        \legacy_if:nTF { @twocolumn }
          {
            \clearpage
            \onecolumn
          }
          { \par \medskip }
        \begingroup
          \small
          \noindent{\bfseries List~of~notes}\par
          \seq_map_inline:Nn \g__zebra_note_types_seq
            { \__zebra_print_note_group:n {##1} }
        \endgroup
      }
  }
```

\zebranewnote

All note types are created with `\zebranewnote`.

```
\NewDocumentCommand \zebranewnote { m m O{} }
  { \__zebra_new_note_type:nnn {#1} {#2} {#3} }
```

(*End of definition for* `\zebranewnote`. *This function is documented on page* 5.)

\zebranotesymbol

Changes the margin symbol of an existing note type.

```
\NewDocumentCommand \zebranotesymbol { m m }
  { \prop_gput:Nnn \g__zebra_note_symbols_prop {#1} {#2} }
```

(*End of definition for* `\zebranotesymbol`. *This function is documented on page* 5.)

\todo
\note
\fixed
\comment
\placeholder

They are defined with `\zebranewnote`. The colors are picked up from the default ones from xcolor.

```
\zebranewnote{todo}{purple}
\zebranewnote{fixed}{teal}
\zebranewnote{comment}{blue}
\zebranewnote{note}{violet}
\zebranewnote{placeholder}{gray}
```

(*End of definition for* `\todo` *and others. These functions are documented on page* 3.)

4.3 Print summary at end of the document

A summary table and a detailed grouped note list are inserted automatically at the end of the document. Each note type with at least one instance is listed with its colour and count, followed by notes grouped by type.

```
\hook_gput_code:nnn { enddocument } { zebra-goodies }
  {
    \bool_if:NT \l__zebra_draft_bool
      {
        \bool_if:NT \l__zebra_listofnotes_bool
          { \__zebra_print_notes: }
      }
  }
\ExplSyntaxOff
⟨/package⟩
```

4.4 Two-column demo

A standalone two-column document used to generate the demo figure included in the documentation. It is extracted automatically by docstrip and compiled during the build.

```
⟨*demo-twocol⟩
\documentclass[twocolumn,a4paper]{article}
\usepackage[margin=2cm]{geometry}
\usepackage{zebra-goodies}
\usepackage{lipsum}
\pagestyle{empty}
\begin{document}
\section{Demo name\comment{revise the name}}
\lipsum[1] \todo[alice]{revise the introduction}

\lipsum[2] \note[bob]{add a reference here}
\lipsum[3] \fixed[carol]{looks good now}
\lipsum[4] \todo[dave]{needs more data}

\lipsum[5] \placeholder[eve]{insert figure}

\lipsum[6] \note[frank]{check the numbers}

\lipsum[7] \comment[tom]{is it you, jerry?}

\lipsum[8] \fixed[heidi]{typo corrected}

You can place \todo[judy]{summarise the findings} anywhere.
\lipsum[9][10-11]
\end{document}
⟨/demo-twocol⟩
```

Change History

v0.1.0
General: Initial public release 1

v0.2.0
General: Fix xcolor conflict 1

v0.3.0
General: Detect command conflicts . . . 1

v0.4.0
General: Show note number for easy reference 1

v0.5.0
General: Use darker color for label . . . 1

v0.6.0
General: Use gray background for label 1

v0.7.0
General: Move to docstrip 2

v0.8.0
General: Fix new note demo 5
`\zebranewnote`: Fix on `\global` for examples 12

v0.8.1
General: Fix doc 5

v0.9.0
General: Fix margin placement, drop tikz, fix counter in moving args, modernize to kernel key-value and NewDocumentCommand 1

v0.9.1
General: Beautify the numbers. 1

v0.9.2
General: Faster. 1

v1.0.0
General: expl3, list of notes and compatibility. 1

v1.1.0
General: Customisable margin symbol, accurate page numbers, code cleanup. 1

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

Zebra Notes

Type	Number
todo	4
fixed	1
comment	1
note	2
placeholder	1
question	1
Total	**10**

List of notes