

zebra — Writing Revision Toolkit*

Ruini Xue[†]

v2.1.0 (2026/06/24)

Abstract

The **zebra** package is a writing revision toolkit. The current release focuses on inline note-taking, with a lightweight set of macros designed to be simple and practical for both solo and collaborative workflows. Five built-in commands—`\todo`, `\note`, `\comment`, `\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 customisable symbol (default: `\textdbend`) in the nearest margin, and summarised with a summary table plus a detailed note list at the end of the document. If `hyperref` is loaded, the detailed list links back to the notes. Passing the `final` option suppresses all notes for production output.



Contents

1	Introduction	2
2	Installation	2
3	Using the package	2
3.1	Package Options	2
3.2	Notes Macros	3
3.3	Two-column Support	5
3.4	Limitations	7
4	Implementation	7
4.1	Package options	7
4.2	Moving-argument deduplication	8
4.3	Main notes macros	13
4.4	Print summary at end of the document	20
4.5	Compatibility shim	20
4.6	Two-column demo	21
	Change History	21

*This package was previously distributed as **zebra-goodies**. The old name still works but will print a deprecation warning. Please update to **zebra**.

[†]Email: xueruini@gmail.com

1 Introduction

zebra is a writing revision toolkit. The current release focuses on inline note-taking. Many note-taking and to-do packages exist for L^AT_EX, 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 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.

2 Installation

zebra is available on CTAN; install it through your L^AT_EX distribution’s package manager. To build from source, run `latexmk zebra.dtx` to extract the package and typeset the documentation in one step.

3 Using the package

Load the package in the preamble with any desired options.

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

3.1 Package Options

- draft** These two options are complementary. Default: **true** (draft mode). 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.
- sort** Controls the order of the detailed note list printed at the end of the document. Default: **none** (document order). **sort=type** groups them by note type.
If **hyperref** is loaded by the document, page numbers in the detailed note list link back to the corresponding notes. **zebra** does not load **hyperref** itself; load it explicitly if links are desired. Without **hyperref**, the list uses ordinary page references.
- unnumbered** Turns off note numbering. Inline markers become `[todo: ...]` (no number), the margin symbol carries no number, the end-of-document detailed note list is omitted (the per-type summary table remains), and `\zebraref` falls back to `\ref`. Useful as a workaround if numbered notes interact badly with a particular class; the summary table count is then an upper-bound estimate. Default: **false**.

3.2 Notes Macros

All note commands share the syntax `\cmd[⟨name⟩]{⟨text⟩}`. Each also has a prefixed alias (e.g. `\zebratodo`) that is always available, regardless of name conflicts. If a short name clashes with another loaded package, **zebra** will *not* overwrite the existing definition; use the prefixed form instead.



<code>\todo</code>	<code>\todo[⟨name⟩]{⟨text⟩}</code>
<code>\zebratodo</code>	<code>\zebratodo[⟨name⟩]{⟨text⟩}</code>

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

The motivation section still feels too vague `\todo{revise the introduction before submission}` and could benefit from a concrete running example to guide the reader through the key ideas step by step.
The motivation section still feels too vague and could benefit from a concrete running example to guide the reader through the key ideas step by step.

The optional argument assigns one or more people to the note. Assignees appear prefixed with `@`, and notes of the same type are numbered sequentially.

The related work section needs more references `\todo[alice]{add two or three citations from the latest survey}` to recent advances in the field. We should also double-check the experimental setup before the camera-ready deadline `\todo[bob, carol]{verify the hyperparameter table against the source code and update any outdated entries and let's check afterwards}`.

1  The related work section needs more references [`TODO 1@alice: add two or three citations from the latest survey`] to recent advances in the field. We should also double-check the experimental
2  setup before the camera-ready deadline [`TODO 2@bob, carol: verify the hyperparameter table against the source code and update any outdated entries and let's check afterwards`].

Notes can appear inside moving arguments such as `\section` and `\caption`. To ensure stable numbering and cross-references, add a `\label` inside the note:

```
\section{Introduction\todo[jerry]{\label{zebra:heading}fix the name}}
\begin{figure}
  \caption{Speed vs distance. \todo{need to insert the figure}}
\end{figure}
```

With a `\label`, the note is counted once regardless of how many times the heading appears (table of contents, running headers, etc.). Notes without a `\label` in moving arguments are still safe but may receive a separate number in each context.

<code>\note</code>	<code>\note[⟨name⟩]{⟨text⟩}</code>
<code>\zebranote</code>	<code>\zebranote[⟨name⟩]{⟨text⟩}</code>

<code>\comment</code>	<code>\comment[⟨name⟩]{⟨text⟩}</code>
<code>\zebracomment</code>	<code>\zebracomment[⟨name⟩]{⟨text⟩}</code>

<code>\fixed</code>	<code>\fixed[⟨name⟩]{⟨text⟩}</code>
<code>\zebrafixed</code>	<code>\zebrafixed[⟨name⟩]{⟨text⟩}</code>








<code>\placeholder</code>	<code>\placeholder[⟨name⟩]{⟨text⟩}</code>
<code>\zebraplaceholder</code>	<code>\zebraplaceholder[⟨name⟩]{⟨text⟩}</code>

These commands share the same syntax and behaviour as `\todo`; they differ only in name and colour, providing semantic distinction for different annotation purposes. Note that `\zebracomment` is used in the example below 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 has been revised `\placeholder[lucy, tom]{good job!}` and the related work comparison strengthened with two additional references. The list of references still needs a second pass `\todo{check bibliography entries for formatting}` before we can finalize the submission.

With those items addressed, the conclusion has been rewritten so the argument flows more naturally from the results. `\fixed[John]{updated the conclusion}` The overall structure now matches the revised outline we agreed on last week. `\note[who]{anything else?}` If not, the draft should be fine.

1  We may want to reorganise [NOTE 1: how should we structure the intro?] this part before the final submission. The experimental setup in Section 2 has already been reviewed by a
1  collaborator [COMMENT 1@tom: the setup description looks clear now]. Results are presented in the following tables and figures, but some of them are still missing.
1  The discussion has been revised [PLACEHOLDER 1@lucy, tom: good job!] and the related work comparison strengthened with two additional references. The list of references still needs a second
3  pass [TODO 3: check bibliography entries for formatting] before we can finalize the submission.
1  With those items addressed, the conclusion has been rewritten so the argument flows more
1  naturally from the results. [FIXED 1@John: updated the conclusion] The overall structure now
2  matches the revised outline we agreed on last week. [NOTE 2@who: anything else?] If not, the draft should be fine.

<code>\zebranewnote</code>	<code>\zebranewnote{⟨note name⟩}{⟨xcolor name⟩}[⟨symbol⟩]</code>
----------------------------	--

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. Any colour expression accepted by `xcolor` works here, including mixes such as `green!50!black` or `teal!60!black`, so a bare `\definecolor`/`\colorlet` beforehand is not required. The optional `⟨symbol⟩` overrides the default margin symbol (`\textdbend`) for this note type only. Per-type symbols can also be changed after loading via `\zebrasetup{symbol/⟨type⟩=⟨symbol⟩}`.

```
\colorlet{mycyan}{cyan!80!black}
\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?](#)]

`\zebraref` `\zebraref{<label>}`

Labels may be placed inside note bodies with the usual `\label` command. Standard `\ref` returns the note number, while `\zebraref` prints the note type together with the number.

```
The motivation section still feels too vague \todo{\label{zebra:intro}revise
the introduction before submission}. The same issue appears again later
\note{see Todo~\ref{zebra:intro} (that is, \zebraref{zebra:intro}) on
p.~\pageref{zebra:intro}}.
```



The motivation section still feels too vague [[TODO 4: revise the introduction before submission](#)].
The same issue appears again later [[NOTE 3: see Todo 4 \(that is, Todo 4\) on p. 5](#)].

As in standard L^AT_EX, labels inside notes are unavailable in `final` mode because the notes themselves are suppressed.

`\zebrasetup` `\zebrasetup{<key=value list>}`

Configures note appearance after loading. Accepted keys:

- `color/<type>=<colour>` — override the colour of a note type.
- `symbol/<type>=<symbol>` — override the margin symbol of a note type.

For example:

```
\zebrasetup{symbol/fixed=\manerrarrow} % like this doc
\zebrasetup{color/todo=red}
```

3.3 Two-column Support

In `twocolumn` documents, the margin 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.

```
\usepackage[paperwidth=21cm,paperheight=15cm,margin=1.1cm]{geometry}
\usepackage{zebra}
\zebrasetup{symbol/comment=$\clubsuit$}
\pagestyle{empty}
\begin{document}
\section{Demo name\comment{revise the name}}
This draft still needs work
\todo[alice]{\label{zebra:intro}revise the introduction}. The
opening paragraph should also explain the main goal more plainly.
Add one more citation here \note[bob]{support this claim}. A
brief roadmap sentence would also make the structure easier to
scan.

The issue raised in Todo~\ref{zebra:intro} still applies in
the conclusion. The table now looks fine
```

```
\fixed[carol]{alignment corrected}, but one figure is still
missing \placeholder[eve]{insert the overview figure}. A short
transition would also help the flow. The middle section should
probably end with a clearer summary sentence before the
discussion begins. That summary can stay compact, but it should
signal why the next section matters.
```

```
Please verify the totals \note[frank]{check the numbers} and
confirm the wording in the last paragraph
\comment[tom]{is this sentence too strong?}. A small typo has
already been fixed \fixed[heidi]{typo corrected}. The ending
should stay short. The final sentence should return to the main
claim rather than repeat background material. You can place
\todo[judy]{summarise the findings} anywhere once the narrative
is stable.
```

```
One more short paragraph is enough to show how \placeholder{wow,
so great!} the markers stay readable in a compact two-column layout.
The example is intentionally small, but it should still look like
a realistic revision pass.\comment{Bye}
\end{document}
```

The code above produces the following output:

1 ♣

1 Demo name[Comment 1: revise the name]

pass.[COMMENT 3: Bye]

♣3

1 🦋

1 🦋

1 🦋

1 🦋

2 🦋

2 ♣

2 🦋

2 🦋

2 🦋

This draft still needs work [TODO 1@alice: revise the introduction]. The opening paragraph should also explain the main goal more plainly. Add one more citation here [NOTE 1@bob: support this claim]. A brief roadmap sentence would also make the structure easier to scan.

The issue raised in Todo 1 still applies in the conclusion. The table now looks fine [FIXED 1@carol: alignment corrected], but one figure is still missing [PLACEHOLDER 1@eve: insert the overview figure]. A short transition would also help the flow. The middle section should probably end with a clearer summary sentence before the discussion begins. That summary can stay compact, but it should signal why the next section matters.

Please verify the totals [NOTE 2@frank: check the numbers] and confirm the wording in the last paragraph [COMMENT 2@tom: is this sentence too strong?]. A small typo has already been fixed [FIXED 2@heidi: typo corrected]. The ending should stay short. The final sentence should return to the main claim rather than repeat background material. You can place [TODO 2@judy: summarise the findings] anywhere once the narrative is stable.

One more short paragraph is enough to show how [PLACEHOLDER 2: wow, so great!] the markers stay readable in a compact two-column layout. The example is intentionally small, but it should still look like a realistic revision

Zebra Notes

Type	Count
todo	2
fixed	2
comment	3
note	2
placeholder	2
Total	11

List of notes

Comment 1.....1

revise the name

Todo 1 @alice.....1

revise the introduction

Note 1 @bob.....1

support this claim

Fixed 1 @carol.....1

alignment corrected

Placeholder 1 @eve.....1

insert the overview figure

Note 2 @frank.....1

check the numbers

Comment 2 @tom.....1

is this sentence too strong?

Fixed 2 @heidi.....1

typo corrected

Todo 2 @judy.....1

3.4 Limitations

- Notes in moving arguments such as `\section` and `\caption` should use `\label` for stable numbering. Unlabelled notes there may be counted again when replayed in contents lists or running heads.
- A labelled note and an unlabelled note with the same type, author, and body may be treated as the same replay. Add a `\label`, vary the text, or avoid identical note bodies.
- Identical unlabelled notes on the same source line, such as `\todo{x}\todo{x}`, collapse to one note.
- In `final` mode, labels inside notes are unavailable. In `unnumbered` mode, the detailed list is omitted and summary counts are only an upper-bound estimate.
- `\zebranewnote` expects command-name-safe note names and a colour expression valid for `xcolor`; invalid values are left to `TEX` or `xcolor` errors.

4 Implementation

```

1 <*package>
2 <@=zebra>

   Version data to start with.
3 \NeedsTeXFormat{LaTeX2e}[2022-06-01]
4 \ProvidesExplPackage{zebra}
5   {2026/06/24}
6   {2.1.0}
7   {Writing Revision Toolkit}

```

4.1 Package options

Package options `draft`, `sort`, and `unnumbered` are created using the kernel key–value interface available since L^AT_EX 2022-06-01. Post-load configuration (`\zebrasetup`) uses a separate `zebra-setup` key family with `color/<type>` and `symbol/<type>` sub-families.

```

8 \bool_new:N \l__zebra_draft_bool
9 \bool_new:N \l__zebra_emit_label_bool
10 \bool_new:N \l__zebra_sort_none_bool
11 \bool_new:N \l__zebra_unnumbered_bool
12 \seq_new:N \g__zebra_note_types_seq
13 \prop_new:N \g__zebra_note_colors_prop
14 \prop_new:N \g__zebra_note_public_alias_prop
15 \int_new:N \g__zebra_note_id_int
16 \tl_new:N \l__zebra_note_target_tl
17 \tl_new:N \l__zebra_note_color_tl
18 \tl_new:N \l__zebra_note_ref_type_tl
19 \tl_new:N \l__zebra_summary_rows_tl
20 \int_new:N \l__zebra_total_notes_int
21 \prop_new:N \g__zebra_note_symbols_prop
22 \tl_new:N \l__zebra_symbol_tl
23 \tl_set:Nn \l__zebra_symbol_tl { \textdbend }
24
25 \msg_new:nnn { zebra } { command-taken }

```

```

26 {
27   The~command~'\iow_char:N\|#1'~is~already~defined.~
28   Use~'\iow_char:N\|zebra#1'~instead.
29 }
30 \msg_new:nnn { zebra } { duplicate-note-label }
31 { Note~label~'#1'~used~by~a~different~note;~second~note~gets~
32   its~own~identity. }
33
34 \prg_new_conditional:Npnn \__zebra_if_package_loaded:n #1 { T , F , TF }
35 {
36   \cs_if_exist:cTF { ver@#1.sty }
37   { \prg_return_true: }
38   { \prg_return_false: }
39 }
40
41 \keys_define:nn { zebra }
42 {
43   draft .bool_set:N = \l__zebra_draft_bool,
44   draft .initial:n = true,
45   final .meta:n = { draft = false },
46   sort .choice:,
47   sort / type .code:n = { \bool_set_false:N \l__zebra_sort_none_bool },
48   sort / none .code:n = { \bool_set_true:N \l__zebra_sort_none_bool },
49   sort .initial:n = none,
50   unnumbered .bool_set:N = \l__zebra_unnumbered_bool,
51   unnumbered .initial:n = false,
52 }
53 \ProcessKeyOptions [ zebra ]
54 \keys_define:nn { zebra-setup / color }
55 {
56   unknown .code:n =
57     { \prop_gput:NVn \g__zebra_note_colors_prop \l_keys_key_str {#1} }
58 }
59 \keys_define:nn { zebra-setup / symbol }
60 {
61   unknown .code:n =
62     { \prop_gput:NVn \g__zebra_note_symbols_prop \l_keys_key_str {#1} }
63 }

```

4.2 Moving-argument deduplication

Notes inside moving arguments (`\section`, `\caption`, etc.) may be processed more than once per compilation pass. Two separate problems are handled independently:

Problem A — `\sbox` re-measurement. `\@makecaption` typesets the caption in an `\sbox` for width measurement, then typesets it again if it is long. Both executions share the same `\inputlineno`, so the *instance key* (`\langle type \rangle | \langle author \rangle | \langle body \rangle | \inputlineno`) catches the replay for both labeled and unlabeled notes. The second execution reuses the first’s allocation and re-renders, so writes that were lost inside the discarded `\sbox` are re-emitted by the actual typesetting pass.

Problem B — TOC/LOF/header replay. The note token is written verbatim to `.toc/.lof/marks` and re-executed in a secondary context with a different `\inputlineno`.

For *labeled* notes the replay is caught by two mechanisms:

1. A *stable key* (`(\type)|\label name`) stored alongside the allocation; a later encounter from marks/headers that still carries the `\label` finds this key and suppresses.
2. A *content signature* (`(\type)|\author)|\sanitised body`) written to the `.aux` file; on the next pass, TOC/LOF encounters whose `\label` was consumed by `\protected@write`'s `\edef` match the signature and suppress.

Unlabeled notes in moving arguments receive independent allocations on each replay (cosmetic duplicate on TOC/LOF/marks); adding `\label` is the recommended fix when stable replay behaviour is required.

Limitation — same-line same-body unlabeled. Two unlabeled notes with identical body on the same source line (e.g. `\todo{x}\todo{x}`) share an instance key with a single `\sbox` replay of the same source location and are indistinguishable from it at the TeX level. Case 1 collapses the second occurrence into the first. Workarounds: split across lines, vary the body, or add `\label`. Notes inside `\fbox`, `tabular` cells, `\parbox`, `minipage`, footnotes, and other inner-mode containers are unaffected — a single source encounter records normally regardless of the surrounding mode.

```

64 \RequirePackage{xcolor}
65 \RequirePackage{marginnote}
66 \cs_new_eq:NN \__zebra_kernel_label:n \label
67 %% -- dedup data structures --
68 %% Maps any key (instance, stable, or content-sig) to the allocation.
69 \prop_new:N \g__zebra_note_target_prop
70 \prop_new:N \g__zebra_note_display_prop
71 %% Content signatures of labeled notes from previous pass (.aux).
72 \prop_new:N \g__zebra_note_sig_known_prop
73 %% Content signatures written this pass (dedup aux writes).
74 \prop_new:N \g__zebra_note_sig_written_prop
75 %% Content signature stored per stable key (for label-conflict detection).
76 \prop_new:N \g__zebra_note_stable_sig_prop
77 %% Group-local set of targets whose \label has been emitted. Sbox
78 %% boundaries reset it (so caption replay re-emits after the sbox is
79 %% discarded), while a same-source-line repeat in body text sees the
80 %% prior emission and skips the redundant \label, avoiding
81 %% multiply-defined labels.
82 \prop_new:N \l__zebra_label_emitted_prop
83 \tl_new:N \l__zebra_note_display_tl
84 \tl_new:N \l__zebra_note_key_tl
85 %% Instance key: unique per source location.
86 \cs_new:Npn \__zebra_instance_key:nnn #1#2#3
87 {
88   \tl_to_str:n {#1}
89   | \tl_to_str:n {#2}
90   | \tl_to_str:n {#3}
91   | \int_eval:n { \tex_inputlineno:D }
92 }
93 %% Content signature: body stringified with ALL \label{...} stripped.
94 %% Matches across body (has labels), TOC (labels consumed by \edef),
95 %% and running heads (which may uppercase the author/body text).

```

```

96 %% The optional \protect prefix covers marks and \protected@write paths.
97 %% Uses replace_all so that multiple labels are all stripped.
98 \cs_new_protected:Npn \__zebra_content_sig:nnnN #1#2#3#4
99 {
100   \tl_set:Nx \l_tmpa_tl { \tl_to_str:n {#3} }
101   \regex_replace_all:nnN
102     { (? : \protect\s* )? \\label\s* \{ [^{}]* \} } { } \l_tmpa_tl
103   \tl_set:Nx #4
104     {
105       \tl_to_str:n {#1}
106       | \str_lowercase:f { \tl_to_str:n {#2} }
107       | \str_lowercase:f { \l_tmpa_tl }
108     }
109 }
110 %% Extract the last \label name from the stringified body
111 %% (the greedy \A.* consumes up to the rightmost \label). Both the
112 %% original encounter and any replay run through this same regex,
113 %% so they agree on which label to use as the stable key.
114 %% Sets #2 to the label name, or clears it if none found.
115 %% The optional \protect prefix covers marks and \protected@write paths.
116 \cs_new_protected:Npn \__zebra_extract_label:nN #1#2
117 {
118   \tl_set:Nx \l_tmpb_tl { \tl_to_str:n {#1} }
119   \tl_set_eq:NN \l_tmpc_tl \l_tmpb_tl
120   \regex_replace_once:nnN
121     { \A .* (? : \protect\s* )? \\label\s* \{ ([^{}]* ) \} .* \Z }
122     { \1 } \l_tmpb_tl
123   \tl_if_eq:NNTF \l_tmpb_tl \l_tmpc_tl
124     { \tl_clear:N #2 }
125     { \tl_set_eq:NN #2 \l_tmpb_tl }
126 }
127 %% Stable key for labeled notes.
128 %% #2 is expected to be already stringified (from regex extraction),
129 %% so no \tl_to_str is applied - otherwise an unexpanded variable
130 %% token would be stringified instead of its value.
131 \cs_new:Npn \__zebra_stable_key:nn #1#2
132 { \tl_to_str:n {#1} | label | #2 }
133 %% Allocate a fresh note: increment the type counter, generate
134 %% a unique hypertarget name, and record the note in the list body.
135 \cs_new_protected:Npn \__zebra_allocate_note:nnn #1#2#3
136 {
137   \int_gincr:c { g__zebra_note_count_#1_int }
138   \tl_set:Nx \l__zebra_note_display_tl { \__zebra_note_count:n {#1} }
139   \int_gincr:N \g__zebra_note_id_int
140   \tl_set:Nx \l__zebra_note_target_tl
141     { zebranote.\int_use:N \g__zebra_note_id_int }
142   \__zebra_record_note:nnnnn
143     {#1}
144     { \l__zebra_note_display_tl }
145     {#2}
146     {#3}
147     { \l__zebra_note_target_tl }
148 }
149 %% --- .aux callback interface (read back on the next compilation) ---

```

```

150 %% \zebra@sig is the ONE macro zebra writes into the .aux file (see
151 %% \__zebra_write_sig:NN below) and re-reads at \begin{document} on the next
152 %% run. It records a content signature together with the originating
153 %% instance key; a later encounter whose instance key differs from the
154 %% stored one is a replay and is suppressed.
155 %%
156 %% Its name MUST stay in the traditional \zebra@... ("@" catcode 11)
157 %% namespace and MUST NOT be renamed to an expl3 internal such as
158 %% \__zebra_sig:nn / \__zebra_sig:nn. The .aux is read back with normal
159 %% catcodes, where "-" and ":" are catcode 12 (other), so an expl3 name
160 %% written verbatim into the .aux would not parse back as a single
161 %% control sequence and the cross-compilation dedup would silently break.
162 %% This is deliberate: do not "tidy" it into the __zebra namespace.
163 %% Re-stringify the arguments for catcode normalisation.
164 \cs_new_protected:Npn \zebra@sig #1#2
165 {
166   \tl_set:Nx \l_tmpa_tl { \tl_to_str:n {#1} }
167   \tl_set:Nx \l_tmpb_tl { \tl_to_str:n {#2} }
168   \prop_gput:NVV \g__zebra_note_sig_known_prop \l_tmpa_tl \l_tmpb_tl
169 }
170 %% Writes one \zebra@sig record to the .aux. Because that token reaches
171 %% the .aux verbatim, the callback above is the only safe spelling - keep
172 %% the two in sync if the payload ever changes.
173 \cs_new_protected:Npn \__zebra_write_sig:NN #1#2
174 {
175   \prop_if_in:NVF \g__zebra_note_sig_written_prop #1
176   {
177     \immediate\write \@auxout
178       { \string\zebra@sig { \tl_use:N #1 } { \tl_use:N #2 } }
179     \prop_gput:NVN \g__zebra_note_sig_written_prop #1 { 1 }
180   }
181 }
182 \cs_if_exist:NTF \dbend
183 {
184   \cs_set_eq:NN \__zebra_saved_dbend: \dbend
185   \cs_undefine:N \dbend
186   \RequirePackage{manfnt}
187   \cs_set_eq:NN \dbend \__zebra_saved_dbend:
188 }
189 { \RequirePackage{manfnt} }
190 \cs_new:Npn \__zebra_pdfstring_note:
191 #1
192 {
193   \str_if_eq:eeT { \tl_to_str:n {#1} } { [ ] }
194   { \__zebra_pdfstring_note_opt:w }
195 }
196 \cs_new:Npn \__zebra_pdfstring_note_opt:w #1 ] #2 { }
197 \cs_new:Npn \__zebra_target:nn #1#2 {#2}
198 \cs_new:Npn \__zebra_link:nn #1#2 {#2}
199 \cs_new:Npn \__zebra_pageref:n #1 { \pageref {#1} }
200 \cs_new:Npn \__zebra_zebra_label_name:n #1 { #1@zebra }
201 \cs_new:Npn \__zebra_zebra_label_type:n #1
202 {
203   \exp_after:wN \use_i:nn

```

```

204     \cs:w r@\_zebra_zebra_label_name:n {#1}\cs_end:
205     { }
206   }
207   \cs_new_protected:Npn \__zebra_write_zebra_label:n #1
208   {
209     \protected@write \@auxout { }
210     {
211       \string\newlabel{\__zebra_zebra_label_name:n {#1}}
212       {{\exp_not:V \l__zebra_note_ref_type_tl}{}}
213     }
214   }
215   \cs_new_protected:Npn \__zebra_note_label:n #1
216   {
217     \__zebra_kernel_label:n {#1}
218     \__zebra_write_zebra_label:n {#1}
219   }
220   \cs_new_protected:Npn \__zebra_zebra_ref:n #1
221   {
222     \cs_if_exist:cTF { r@\_zebra_zebra_label_name:n {#1} }
223     { \__zebra_zebra_label_type:n {#1}~\ref{#1} }
224     { \ref{#1} }
225   }
226   \NewDocumentCommand \zebraref { m }
227   {
228     \bool_if:NTF \l__zebra_unnumbered_bool
229     { \ref{#1} }
230     { \__zebra_zebra_ref:n {#1} }
231   }
232   \cs_new_protected:Npn \__zebra_apply_pdfstring_defs:
233   {
234     \pdfstringdefDisableCommands
235     {
236       \cs_set:Npn \zebraref ##1 { \ref{##1} }
237       \seq_map_inline:Nn \g__zebra_note_types_seq
238       {
239         \cs_set_eq:cN { zebra##1 } \__zebra_pdfstring_note:
240         \prop_if_in:NnT \g__zebra_note_public_alias_prop { ##1 }
241         { \cs_set_eq:cN { ##1 } \__zebra_pdfstring_note: }
242       }
243     }
244   }
245   \cs_new_protected:Npn \__zebra_setup_hyperref:
246   {
247     \cs_set:Npn \__zebra_target:nn ##1##2 {##2}
248     \cs_set:Npn \__zebra_link:nn ##1##2 {##2}
249     \cs_set:Npn \__zebra_pageref:n ##1 { \pageref {##1} }
250     \__zebra_if_package_loaded:nT { hyperref }
251     {
252       \cs_set:Npn \__zebra_pageref:n ##1 { \pageref* {##1} }
253       \cs_set:Npn \__zebra_target:nn ##1##2 { \hypertarget{##1}{##2} }
254       \cs_set:Npn \__zebra_link:nn ##1##2 { \hyperlink{##1}{##2} }
255       \__zebra_apply_pdfstring_defs:
256     }
257   }

```

```

258 \hook_gput_code:nnn { begindocument } { zebra }
259 { \__zebra_setup_hyperref: }

```

4.3 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 single-column mode `\marginnote` is used with the right-margin variant as default. In twocolumn mode `\marginpar` positions the symbol close to the text column reliably (it picks side from `\if@firstcolumn`, not from a `.aux` round trip), so we keep that path for the body. We must, however, route around the contexts where `\marginpar` would crash with “Float(s) lost”: caption sboxes (inner mode), caption parboxes (signalled by `\@capytype`), and the wide `\vbox` that `\@topnewpage` builds for `\title` (signalled by `\hsize > \columnwidth`). Those contexts fall back to `\marginnote`, where we additionally replicate `\@marginparreset` (`\@parboxrestore + \normalfont\normalsize`) so the marker does not inherit a `\Huge` font from `\title`.

```

260 \cs_new_protected:Npn \__zebra_margin_note:nn #1#2
261 {
262   \legacy_if:nTF { @twocolumn }
263   {
264     \bool_lazy_or:nnTF
265     { \mode_if_inner_p: }
266     {
267       \bool_lazy_or_p:nn
268       { \cs_if_exist_p:N \@capytype }
269       { \dim_compare_p:nNn { \hsize } > { \columnwidth } }
270     }
271     {
272       \marginnote
273       [ { \@parboxrestore \normalfont \normalsize #1 } ]
274       { \@parboxrestore \normalfont \normalsize #2 }
275     }
276     {
277       \marginpar
278       [ { \makebox[\marginparwidth][r]{#1} } ]
279       { \makebox[\marginparwidth][l]{#2} }
280     }
281   }
282   { \marginnote[#1]{#2} }
283 }
284 \cs_new:Npn \__zebra_prepend:nn #1#2
285 { \tl_if_blank:nTF {#2} {} {#1#2} }
286 \cs_new:Npn \__zebra_capitalize_type:n #1
287 { \text_uppercase:n { \tl_head:n {#1} } \tl_tail:n {#1} }
288 \cs_new:Npn \__zebra_note_count:n #1
289 { \int_use:c { g__zebra_note_count_#1_int } }
290 \cs_new:Npn \__zebra_note_color:n #1
291 { \prop_item:Nn \g__zebra_note_colors_prop {#1} }
292 \cs_new:Npn \__zebra_note_symbol:n #1
293 {

```

```

294 \prop_if_in:NnTF \g__zebra_note_symbols_prop {#1}
295 { \prop_item:Nn \g__zebra_note_symbols_prop {#1} }
296 { \l__zebra_symbol_tl }
297 }
298 \cs_new_protected:Npn \__zebra_new_listbody:n #1
299 { \tl_new:c { g__zebra_listbody_#1_tl } }
300 \tl_new:N \g__zebra_listbody_all_tl
301 \cs_new:Npn \__zebra_use_listbody:n #1
302 { \tl_use:c { g__zebra_listbody_#1_tl } }
303 \cs_new_protected:Npn \__zebra_record_note:nnnnn #1#2#3#4#5
304 {
305   \tl_gput_right:cx
306   {
307     \bool_if:NTF \l__zebra_sort_none_bool
308     { g__zebra_listbody_all_tl }
309     { g__zebra_listbody_#1_tl }
310   }
311   {
312     \exp_not:N \__zebra_list_entry:nnnnn
313     { \exp_not:n {#1} }
314     {#2}
315     { \exp_not:n {#3} }
316     { \exp_not:n {#4} }
317     {#5}
318   }
319 }
320 %% \__zebra_note_unnumbered:nnn {type}{author}{body}
321 %% Fast path for the \opt{unnumbered} option: skip dedup, .aux
322 %% signatures and list registration entirely. The counter is still
323 %% bumped on every call (so the summary table reflects rough counts;
324 %% notes inside captions/TOC/marks may be over-counted, which is the
325 %% accepted trade-off for cutting out the fragile bookkeeping).
326 \cs_new_protected:Npn \__zebra_note_unnumbered:nnn #1#2#3
327 {
328   \bool_if:NT \l__zebra_draft_bool
329   {
330     \tl_set:Nx \l__zebra_note_color_tl { \__zebra_note_color:n {#1} }
331     \int_gincr:c { g__zebra_note_count_#1_int }
332     \__zebra_render_note:nnn {#1} {#2} {#3}
333   }
334 }
335 %% \__zebra_note:nnn {type}{author}{body}
336 %% Main entry point: dispatch to the lightweight unnumbered path or to
337 %% the numbered path with deduplication/bookkeeping.
338 \cs_new_protected:Npn \__zebra_note:nnn #1#2#3
339 {
340   \bool_if:NTF \l__zebra_unnumbered_bool
341   { \__zebra_note_unnumbered:nnn {#1} {#2} {#3} }
342   { \__zebra_note_numbered:nnn {#1} {#2} {#3} }
343 }
344 %% Numbered path. Four cases:
345 %% Case 1 - sbox reuse: instance_key found → reuse, render
346 %% Case 2 - stable key: label found, stable_key in prop → suppress
347 %% Case 3 - content sig: sig in .aux data → suppress

```

```

348 %% Case 4 - new note: allocate, render
349 %% Case 1 also catches an unlabeled same-source-line same-body pair
350 %% (\todo{x}\todo{x}); the second is collapsed. Two such notes are
351 %% indistinguishable at TeX level from a single note replayed by an
352 %% sbox/moving argument, so this collision is documented and resolved
353 %% by adding \label or splitting the line.
354 \cs_new_protected:Npn \__zebra_note_numbered:nnn #1#2#3
355 {
356   \bool_if:NT \l__zebra_draft_bool
357   {
358     \tl_set:Nx \l__zebra_note_color_tl { \__zebra_note_color:n {#1} }
359     %% Case 1: sbox reuse (same \inputlineno)
360     \tl_set:Nx \l__zebra_note_key_tl
361     { \__zebra_instance_key:nnn {#1} {#2} {#3} }
362     \prop_get:NVNTF \g__zebra_note_target_prop \l__zebra_note_key_tl
363     \l__zebra_note_target_tl
364     {
365       \prop_get:NVN \g__zebra_note_display_prop \l__zebra_note_key_tl
366       \l__zebra_note_display_tl
367       \__zebra_render_note:nnn {#1} {#2} {#3}
368     }
369   }
370   {
371     %% Extract label and compute content signature
372     \__zebra_extract_label:nN {#3} \l_tmpb_tl
373     \__zebra_content_sig:nnnN {#1} {#2} {#3} \l_tmpa_tl
374     %% Case 2: stable-key suppress (labeled, marks/headers).
375     %% If the stable key exists AND the content signature
376     %% matches, this encounter is a replay → suppress.
377     %% Different content signature = label reuse → warn and
378     %% let Case 4 allocate independently.
379     \bool_set_false:N \l_tmpa_bool
380     \tl_if_empty:NF \l_tmpb_tl
381     {
382       \tl_set:Nx \l__zebra_note_key_tl
383       { \__zebra_stable_key:nn {#1} { \l_tmpb_tl } }
384       \prop_get:NVNT \g__zebra_note_stable_sig_prop
385       \l__zebra_note_key_tl \l_tmpc_tl
386       {
387         \tl_if_eq:NNTF \l_tmpa_tl \l_tmpc_tl
388         { \bool_set_true:N \l_tmpa_bool }
389         {
390           \msg_warning:nnV { zebra }
391           { duplicate-note-label } \l_tmpb_tl
392         }
393       }
394     }
395     %% Case 3: content-sig suppress (TOC/LOF/marks replay).
396     %% Only unlabeled current encounters may be suppressed by
397     %% content signature; labeled encounters already have a
398     %% strong identity via the stable key. Suppress only if
399     %% the stored instance key differs from the current one -
400     %% same key means it is the original note, not a replay.
401     \bool_if:NF \l_tmpa_bool
402     {

```

```

402 \tl_if_empty:NT \l_tmpb_tl
403 {
404   \prop_get:NVNT \g__zebra_note_sig_known_prop
405   \l_tmpa_tl \l_tmpc_tl
406   {
407     \tl_set:Nx \l_tmpd_tl
408     { \__zebra_instance_key:nnn {#1} {#2} {#3} }
409     \tl_if_eq:NNT \l_tmpc_tl \l_tmpd_tl
410     { \bool_set_true:N \l_tmpa_bool }
411   }
412 }
413 }
414 \bool_if:NF \l_tmpa_bool
415 {
416   %% Case 4: new note - allocate and render
417   \tl_set:Nx \l__zebra_note_key_tl
418   { \__zebra_instance_key:nnn {#1} {#2} {#3} }
419   \__zebra_allocate_note:nnn {#1} {#2} {#3}
420   \prop_gput:NVV \g__zebra_note_target_prop
421   \l__zebra_note_key_tl \l__zebra_note_target_tl
422   \prop_gput:NVV \g__zebra_note_display_prop
423   \l__zebra_note_key_tl \l__zebra_note_display_tl
424   %% For labeled notes: register stable key + write sig,
425   %% but only if the stable key is not already claimed by
426   %% an earlier note (label-conflict case).
427   \tl_if_empty:NF \l_tmpb_tl
428   {
429     \tl_set:Nx \l__zebra_note_key_tl
430     { \__zebra_stable_key:nn {#1} { \l_tmpb_tl } }
431     \prop_if_in:NVF \g__zebra_note_stable_sig_prop
432     \l__zebra_note_key_tl
433     {
434       \prop_gput:NVV \g__zebra_note_target_prop
435       \l__zebra_note_key_tl \l__zebra_note_target_tl
436       \prop_gput:NVV \g__zebra_note_display_prop
437       \l__zebra_note_key_tl \l__zebra_note_display_tl
438       \prop_gput:NVV \g__zebra_note_stable_sig_prop
439       \l__zebra_note_key_tl \l_tmpa_tl
440       \tl_set:Nx \l_tmpc_tl
441       { \__zebra_instance_key:nnn {#1} {#2} {#3} }
442       \__zebra_write_sig:NN \l_tmpa_tl \l_tmpc_tl
443     }
444   }
445   \__zebra_render_note:nnn {#1} {#2} {#3}
446 }
447 %% Cases 2-3: suppress - no output
448 }
449 }
450 }
451 %% Full render: hypertarget, target label, margin note, inline text.
452 %% In \opt{unnumbered} mode the label/hypertarget setup is skipped so a
453 %% \cs{label} inside the note body falls through to the surrounding
454 %% \cs{@currentlabel} (e.g.\ the section number), and \cs{zebraref}
455 %% degrades cleanly to \cs{ref}.

```



```

456 \cs_new_protected:Npn \__zebra_render_note:nnn #1#2#3
457 {
458   %% Decide outside render's group whether to emit the kernel label.
459   %% The flag lives in the caller's group: sbox/parbox boundaries reset
460   %% it (so caption replay re-emits), but a same-line repeat in body
461   %% text sees the prior emission and skips the redundant \label.
462   \bool_set_false:N \l__zebra_emit_label_bool
463   \bool_if:NF \l__zebra_unnumbered_bool
464   {
465     \prop_if_in:NVF \l__zebra_label_emitted_prop \l__zebra_note_target_tl
466     {
467       \bool_set_true:N \l__zebra_emit_label_bool
468       \prop_put:NVn \l__zebra_label_emitted_prop
469         \l__zebra_note_target_tl { 1 }
470     }
471   }
472   \group_begin:
473   \bool_if:NF \l__zebra_unnumbered_bool
474   {
475     \protected@edef \@currentlabel { \l__zebra_note_display_tl }
476     \__zebra_if_package_loaded:nT { hyperref }
477     { \tl_set:Nx \@currentHref { \l__zebra_note_target_tl } }
478     \tl_set:Nx \l__zebra_note_ref_type_tl
479       { \__zebra_capitalize_type:n {#1} }
480     \__zebra_target:nn { \l__zebra_note_target_tl } {}
481     \bool_if:NT \l__zebra_emit_label_bool
482     { \exp_args:NW \__zebra_kernel_label:n \l__zebra_note_target_tl }
483   }
484   \__zebra_margin_note:nn
485     {\textcolor{\l__zebra_note_color_tl}{%
486       \bool_if:NF \l__zebra_unnumbered_bool
487       { {\bfseries\l__zebra_note_display_tl}\kern1pt }%
488       \__zebra_note_symbol:n {#1}}}%
489     {\textcolor{\l__zebra_note_color_tl}{%
490       \__zebra_note_symbol:n {#1}%
491       \bool_if:NF \l__zebra_unnumbered_bool
492       { \kern1pt {\bfseries\l__zebra_note_display_tl} }}}}%
493   \bool_if:NF \l__zebra_unnumbered_bool
494   { \cs_set_eq:NN \label \__zebra_note_label:n }
495   \textcolor{\l__zebra_note_color_tl}{[\colorbox[gray]{0.97}{%
496     \textcolor{\l__zebra_note_color_tl !70!black}{%
497       \textsc{\MakeLowercase{\MakeUppercase#1}}}%
498     \bool_if:NF \l__zebra_unnumbered_bool
499     { ~\l__zebra_note_display_tl }%
500     \texttt{\__zebra_prepend:nn {@}{#2}}:}} #3}}%
501   \group_end:
502 }
503 \cs_new_protected:Npn \__zebra_new_note_type:nn #1#2
504 { \__zebra_new_note_type:nnn {#1} {#2} {} }
505 \cs_new_protected:Npn \__zebra_new_note_type:nnn #1#2#3
506 {
507   \seq_gput_right:Nn \g__zebra_note_types_seq {#1}
508   \prop_if_in:NnF \g__zebra_note_colors_prop {#1}
509   { \prop_gput:Nnn \g__zebra_note_colors_prop {#1} {#2} }

```

```

510 \tl_if_blank:nF {#3}
511 {
512   \prop_if_in:NnF \g__zebra_note_symbols_prop {#1}
513   { \prop_gput:Nnn \g__zebra_note_symbols_prop {#1} {#3} }
514 }
515 \int_new:c { g__zebra_note_count_#1_int }
516 \__zebra_new_listbody:n {#1}
517 \exp_args:Nc \NewDocumentCommand { zebra#1 } { 0{ } m }
518 { \__zebra_note:nnn {#1}{##1}{##2} }
519 \cs_if_exist:cTF {#1}
520 { \msg_warning:nnn { zebra } { command-taken } {#1} }
521 {
522   \cs_set_eq:cc {#1} {zebra#1}
523   \prop_gput:Nnn \g__zebra_note_public_alias_prop {#1} { true }
524 }
525 \__zebra_if_package_loaded:nT { hyperref }
526 { \__zebra_apply_pdfstring_defs: }
527 }
528 \cs_new_protected:Npn \__zebra_list_entry:nnnnn #1#2#3#4#5
529 {
530   \par\noindent
531   \textcolor{\__zebra_note_color:n {#1}}{
532     \textbf{\__zebra_capitalize_type:n {#1}~#2}%
533     \tl_if_blank:nF {#3} { \enspace \texttt{\__zebra_prepend:nn {0}{#3}} } }%
534   \nobreak\dotfill
535   \__zebra_link:nn {#5} { \__zebra_pageref:n {#5} }%
536   \par
537   \begingroup
538     \leftskip=2em
539     \rightskip=2em
540     \parindent=0pt
541     \cs_set_eq:NN \label \use_none:n
542     #4\par
543   \endgroup
544 }
545 \cs_new_protected:Npn \__zebra_print_note_group:n #1
546 {
547   \int_compare:nNnT { \__zebra_note_count:n {#1} } > { 0 }
548   {
549     \par\medskip
550     \__zebra_use_listbody:n {#1}
551   }
552 }
553 \cs_new_protected:Npn \__zebra_print_notes_inorder:
554 {
555   \tl_if_empty:NF \g__zebra_listbody_all_tl
556   { \par\medskip \tl_use:N \g__zebra_listbody_all_tl }
557 }
558 \cs_new_protected:Npn \__zebra_summary_row:n #1
559 {
560   \int_compare:nNnT { \__zebra_note_count:n {#1} } > { 0 }
561   {
562     \int_add:Nn \l__zebra_total_notes_int { \__zebra_note_count:n {#1} }
563     \tl_put_right:Nx \l__zebra_summary_rows_tl

```

```

564     {
565         \exp_not:N \textcolor
566         { \_zebra_note_color:n {#1} }
567         {#1}
568         \exp_not:N &
569         \_zebra_note_count:n {#1}
570         \exp_not:N \\
571     }
572 }
573 }
574 \cs_new_protected:Npn \_zebra_print_notes:
575 {
576     \tl_clear:N \l_zebra_summary_rows_tl
577     \int_zero:N \l_zebra_total_notes_int
578     \seq_map_inline:Nn \g_zebra_note_types_seq
579     { \_zebra_summary_row:n {##1} }
580     \tl_if_empty:NF \l_zebra_summary_rows_tl
581     {
582         \par\nobreak
583         \noindent\dotfill\par\medskip
584         \nobreak
585         \noindent\textbf{\Large Zebra~Notes}
586         \par \medskip
587         \begin{center}
588             \begin{tabular}{lr}
589                 \hline
590                 \textbf{Type} & \textbf{Count} \\ \hline
591                 \tl_use:N \l_zebra_summary_rows_tl
592                 \hline
593                 \textbf{Total} & \int_use:N \l_zebra_total_notes_int \\ \hline
594                 \hline
595             \end{tabular}
596         \end{center}
597         \bool_if:NF \l_zebra_unnumbered_bool
598         {
599             \par \medskip
600             \begin{group}
601                 \small
602                 \noindent{\bfseries List~of~notes}\par
603                 \nobreak
604                 \bool_if:NTF \l_zebra_sort_none_bool
605                 { \_zebra_print_notes_inorder: }
606                 {
607                     \seq_map_inline:Nn \g_zebra_note_types_seq
608                     { \_zebra_print_note_group:n {##1} }
609                 }
610             \end{group}
611         }
612     }
613 }

```

\zebranewnote All note types are created with \zebranewnote.

```

614 \NewDocumentCommand \zebranewnote { m m O{} }
615 { \_zebra_new_note_type:nnn {#1} {#2} {#3} }

```

(End of definition for `\zebranewnote`. This function is documented on page 4.)

`\zebrasetup` Applies configuration keys after loading using the `zebra-setup` key family.

```
616 \NewDocumentCommand \zebrasetup { m }
617   { \keys_set:nn { zebra-setup } {#1} }
```

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

`\todo` Built-in note types, defined with `\zebranewnote`.

```
\note 618 \zebranewnote{todo}{purple}
\fixed 619 \zebranewnote{fixed}{green!50!black}
\comment 620 \zebranewnote{comment}{blue}
\placeholder 621 \zebranewnote{note}{violet}
622 \zebranewnote{placeholder}{gray}
```

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

4.4 Print summary at end of the document

A summary table and a detailed 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 in document order or grouped by type.

```
623 %% At end of document: print the note summary and list.
624 %% Content signatures are written to .aux inline (at allocation time),
625 %% so no additional end-of-document aux writes are needed.
626 \hook_gput_code:nnn { enddocument } { zebra }
627 {
628   \bool_if:NT \l__zebra_draft_bool
629     { \__zebra_print_notes: }
630 }
631 \ExplSyntaxOff
632 \end{package}
```

4.5 Compatibility shim

The old package name `zebra-goodies` is supported via a thin wrapper that loads `zebra` and prints a deprecation warning.

```
633 \compat
634 \NeedsTeXFormat{LaTeX2e}
635 \ProvidesPackage{zebra-goodies}
636   [2026/06/24 v2.1.0 Deprecated: use zebra instead]
637 \PackageWarningNoLine{zebra-goodies}
638   {Package 'zebra-goodies' is deprecated.\MessageBreak
639     Use \string\usepackage{zebra} instead}
640 \RequirePackageWithOptions{zebra}
641 \end{compat}
```

4.6 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.

```

642 <*demo-twocol>
643 \documentclass[twocolumn]{article}
644 \usepackage[paperwidth=21cm,paperheight=15cm,margin=1.1cm]{geometry}
645 \usepackage{zebra}
646 \zebrasetup{symbol/comment=$\clubsuit$}
647 \pagestyle{empty}
648 \begin{document}
649 \section{Demo name\comment{revise the name}}
650 This draft still needs work
651 \todo[alice]{\label{zebra:intro}revise the introduction}. The
652 opening paragraph should also explain the main goal more plainly.
653 Add one more citation here \note[bob]{support this claim}. A
654 brief roadmap sentence would also make the structure easier to
655 scan.
656
657 The issue raised in Todo~\ref{zebra:intro} still applies in
658 the conclusion. The table now looks fine
659 \fixed[carol]{alignment corrected}, but one figure is still
660 missing \placeholder[eve]{insert the overview figure}. A short
661 transition would also help the flow. The middle section should
662 probably end with a clearer summary sentence before the
663 discussion begins. That summary can stay compact, but it should
664 signal why the next section matters.
665
666 Please verify the totals \note[frank]{check the numbers} and
667 confirm the wording in the last paragraph
668 \comment[tom]{is this sentence too strong?}. A small typo has
669 already been fixed \fixed[heidi]{typo corrected}. The ending
670 should stay short. The final sentence should return to the main
671 claim rather than repeat background material. You can place
672 \todo[judy]{summarise the findings} anywhere once the narrative
673 is stable.
674
675 One more short paragraph is enough to show how \placeholder{wow,
676 so great!} the markers stay readable in a compact two-column layout.
677 The example is intentionally small, but it should still look like
678 a realistic revision pass.\comment{Bye}
679 \end{document}
680 </demo-twocol>

```

Change History

v0.1.0		v0.3.0
General: Initial public release	1	General: Detect command conflicts . . . 1
v0.2.0		v0.4.0
General: Fix xcolor conflict	1	General: Show note number for easy

reference	1	v1.4.0	
v0.5.0		General: Support note labels via	
General: Use darker color for label . . .	1	\label, \ref, and \zebraref. . . .	1
v0.6.0		v1.5.0	
General: Use gray background for label	1	General: Fix notes numbering in	
v0.7.0		moving arguments.	1
General: Move to docstrip	2	v1.6.0	
v0.8.0		General: Numbering in moving	
General: Fix new note demo	4	arguments is hard.	1
\zebranewnote: Fix on \global for		v1.7.0	
examples	19	General: Robust margin notes in	
v0.8.1		twocolumn captions, titles and	
General: Fix doc	4	headings.	1
v0.9.0		v1.8.0	
General: Fix legacy bugs and improve		General: Add unnumbered option for	
implementation	1	lightweight mode.	1
v0.9.1		Fix margin symbol distance.	1
General: Beautify the numbers.	1	Stabilise labeled notes replayed	
v0.9.2		through uppercased running heads. . .	1
General: Faster.	1	v1.8.1	
v1.0.0		General: Drop note bodies from PDF	
General: expl3 , list of notes and		bookmarks.	1
compatibility.	1	Fix multiply-defined labels for	
v1.1.0		repeated unlabeled notes.	1
General: Customisable margin symbol,		Keep labeled notes with shared	
accurate page numbers, code		content distinct.	1
cleanup.	1	Silence pass-1 warning from	
v1.1.1		\zebraref.	1
General: Per-type color/symbol keys,		v1.9.0	
\zebrasetup.	1	General: Document known limitations. .	1
v1.2.0		v2.0.0	
General: Simplify key architecture. . . .	1	General: Stop loading hyperref and	
v1.3.0		microtype	1
General: Rename package to zebra	1	v2.1.0	
Rename the microtype expansion		General: Accept any xcolor colour	
option to font-expansion	1	expression in \zebranewnote. . . .	1
Rename the page-link option to			
pagelinks/nopagelinks	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.

Symbols	Numbers
\ \	27, 28, 102, 121, 570, 590, 593
\{	102, 121
\}	102, 121
_	454
\1	122
	A
\A	111, 121

B	
<code>\begin</code>	151, 587, 588, 648
<code>\begingroup</code>	537, 600
<code>\bfseries</code>	487, 492, 602
bool commands:	
<code>\bool_if:NTF</code>	228, 307,
328, 340, 356, 400, 414, 463, 473,	
481, 486, 491, 493, 498, 597, 604, 628	
<code>\bool_lazy_or:nnTF</code>	264
<code>\bool_lazy_or_p:nn</code>	267
<code>\bool_new:N</code>	8, 9, 10, 11
<code>\bool_set_false:N</code>	47, 378, 462
<code>\bool_set_true:N</code>	48, 387, 410, 467
<code>\l_tmpa_bool</code> ..	378, 387, 400, 410, 414
C	
<code>\caption</code>	3, 7, 8
<code>\clubsuit</code>	646
<code>\cmd</code>	3
<code>\colorbox</code>	495
<code>\colorlet</code>	4
<code>\columnwidth</code>	13, 269
<code>\comment</code>	1, 3, 4, 618, 649, 668, 678
<code>\cs</code>	453, 454, 455
cs commands:	
<code>\cs:w</code>	204
<code>\cs_end:</code>	204
<code>\cs_if_exist:NTF</code>	36, 182, 222, 519
<code>\cs_if_exist_p:N</code>	268
<code>\cs_new:Npn</code>	
86, 131, 190, 196, 197, 198, 199,	
200, 201, 284, 286, 288, 290, 292, 301	
<code>\cs_new_eq:NN</code>	66
<code>\cs_new_protected:Npn</code> ..	98, 116,
135, 164, 173, 207, 215, 220, 232,	
245, 260, 298, 303, 326, 338, 354,	
456, 503, 505, 528, 545, 553, 558, 574	
<code>\cs_set:Npn</code>	
236, 247, 248, 249, 252, 253, 254	
<code>\cs_set_eq:NN</code>	
184, 187, 239, 241, 494, 522, 541	
<code>\cs_undefine:N</code>	185
D	
<code>\dbend</code>	182, 184, 185, 187
<code>\definecolor</code>	4
dim commands:	
<code>\dim_compare_p:nNn</code>	269
<code>\documentclass</code>	643
<code>\dotfill</code>	534, 583
<code>draft</code> (option)	2
E	
<code>\edef</code>	9, 94
<code>\end</code>	595, 596, 679
<code>\endgroup</code>	543, 610
<code>\enspace</code>	533
exp commands:	
<code>\exp_after:wN</code>	203
<code>\exp_args:Nc</code>	517
<code>\exp_args:NV</code>	482
<code>\exp_not:N</code>	312, 565, 568, 570
<code>\exp_not:n</code>	212, 313, 315, 316
<code>\ExplSyntaxOff</code>	631
F	
<code>\fbox</code>	9
<code>final</code> (option)	2
<code>\fixed</code>	1, 4, 618, 659, 669
G	
<code>\global</code>	22
group commands:	
<code>\group_begin:</code>	472
<code>\group_end:</code>	501
H	
<code>\hline</code>	589, 590, 592, 594
hook commands:	
<code>\hook_gput_code:nnn</code>	258, 626
<code>\hsize</code>	13, 269
<code>\Huge</code>	13
<code>\hyperlink</code>	254
<code>\hypertarget</code>	253
I	
<code>\immediate</code>	177
<code>\inputlineno</code>	8, 359
int commands:	
<code>\int_add:Nn</code>	562
<code>\int_compare:nNnTF</code>	547, 560
<code>\int_eval:n</code>	91
<code>\int_gincr:N</code>	137, 139, 331
<code>\int_new:N</code>	15, 20, 515
<code>\int_use:N</code>	141, 289, 593
<code>\int_zero:N</code>	577
iow commands:	
<code>\iow_char:N</code>	27, 28
K	
<code>\kern</code>	487, 492
keys commands:	
<code>\keys_define:nn</code>	41, 54, 59
<code>\l_keys_key_str</code>	57, 62
<code>\keys_set:nn</code>	617
L	
<code>\label</code>	3, 5, 7, 9, 22, 66, 77,
	80, 93, 110, 111, 353, 461, 494, 541, 651

<code>\Large</code>	585	<code>\prop_get:NnNTF</code>	362, 383, 404
<code>\leftskip</code>	538	<code>\prop_gput:Nnn</code> ..	57, 62, 168, 179, 420, 422, 434, 436, 438, 509, 513, 523
legacy commands:		<code>\prop_if_in:NnTF</code>	175, 240, 294, 431, 465, 508, 512
<code>\legacy_if:nTF</code>	262	<code>\prop_item:Nn</code>	291, 295
M			
<code>\makebox</code>	278, 279	<code>\prop_new:N</code>	13, 14, 21, 69, 70, 72, 74, 76, 82
<code>\MakeLowercase</code>	497	<code>\prop_put:Nnn</code>	468
<code>\MakeUppercase</code>	497	<code>\protect</code>	96, 115
<code>\marginnote</code>	13, 272, 282	<code>\ProvidesExplPackage</code>	4
<code>\marginpar</code>	13, 277	<code>\ProvidesPackage</code>	635
<code>\marginparwidth</code>	278, 279	Q	
<code>\medskip</code>	549, 556, 583, 586, 599	<code>\question</code>	4
<code>\MessageBreak</code>	638	R	
mode commands:		<code>\ref</code>	2, 5, 22, 223, 224, 229, 236, 657
<code>\mode_if_inner_p:</code>	265	regex commands:	
msg commands:		<code>\regex_replace_all:nnN</code>	101
<code>\msg_new:nnn</code>	25, 30	<code>\regex_replace_once:nnN</code>	120
<code>\msg_warning:nnn</code>	389, 520	<code>\RequirePackage</code>	64, 65, 186, 189
N		<code>\RequirePackageWithOptions</code>	640
<code>\NeedsTeXFormat</code>	3, 634	<code>\rightskip</code>	539
<code>\NewDocumentCommand</code> ..	226, 517, 614, 616	S	
<code>\newlabel</code>	211	<code>\s</code>	102, 121
<code>\nobreak</code>	534, 582, 584, 603	<code>\sbox</code>	8, 9
<code>\noindent</code>	530, 583, 585, 602	<code>\section</code>	3, 7, 8, 649
<code>\normalfont</code>	13, 273, 274	seq commands:	
<code>\normalsize</code>	13, 273, 274	<code>\seq_gput_right:Nn</code>	507
<code>\note</code>	1, 3, 618, 653, 666	<code>\seq_map_inline:Nn</code>	237, 578, 607
O		<code>\seq_new:N</code>	12
<code>\opt</code>	321, 452	<code>\small</code>	601
options:		sort (option)	2
draft	2	str commands:	
final	2	<code>\str_if_eq:nnTF</code>	193
sort	2	<code>\str_lowercase:n</code>	106, 107
unnumbered	2	<code>\string</code>	178, 211, 639
P			
<code>\PackageWarningNoLine</code>	637	T	
<code>\pageref</code>	199, 249, 252	TeX and L ^A T _E X 2 _ε commands:	
<code>\pagestyle</code>	647	<code>\@auxout</code>	177, 209
<code>\par</code>	530, 536, 542, 549, 556, 582, 583, 586, 599, 602	<code>\@capttype</code>	13, 268
<code>\parbox</code>	9	<code>\@currentHref</code>	477
<code>\parindent</code>	540	<code>\@currentlabel</code>	475
<code>\pdfstringdefDisableCommands</code>	234	<code>\@makecaption</code>	8
<code>\placeholder</code>	1, 4, 618, 660, 675	<code>\@marginparreset</code>	13
prg commands:		<code>\@parboxrestore</code>	13, 273, 274
<code>\prg_new_conditional:Npnn</code>	34	<code>\@topnewpage</code>	13
<code>\prg_return_false:</code>	38	<code>\if@firstcolumn</code>	13
<code>\prg_return_true:</code>	37	<code>\protected@edef</code>	475
<code>\ProcessKeyOptions</code>	53	<code>\protected@write</code>	9, 96, 115, 209
prop commands:		<code>\zebra@</code>	156
<code>\prop_get:NnN</code>	365	<code>\zebra@sig</code>	150, 164, 170, 178

\g_zebra_note_sig_known_prop . . .	_zebra_sig:nn	158
. 72, 168, 404	\l_zebra_sort_none_bool	
\g_zebra_note_sig_written_prop 10, 47, 48, 307, 604	
. 74, 175, 179	_zebra_stable_key:nn	131, 382, 430
\g_zebra_note_stable_sig_prop . .	_zebra_summary_row:n	558, 579
. 76, 383, 431, 438	\l_zebra_summary_rows_tl	
_zebra_note_symbol:n 19, 563, 576, 580, 591	
292, 488, 490	\l_zebra_symbol_tl	22, 23, 296
\g_zebra_note_symbols_prop	_zebra_target:nn	197, 247, 253, 480
. 21, 62, 294, 295, 512, 513	\l_zebra_total_notes_int	
\g_zebra_note_target_prop 20, 562, 577, 593	
. 69, 362, 420, 434	\l_zebra_unnumbered_bool	
\l_zebra_note_target_tl 11, 50, 228,	
. 16, 140, 147,	340, 463, 473, 486, 491, 493, 498, 597	
363, 421, 435, 465, 469, 477, 480, 482	_zebra_use_listbody:n	301, 550
\g_zebra_note_types_seq	_zebra_write_sig:NN	151, 173, 442
. 12, 237, 507, 578, 607	_zebra_write_zebra_label:n	207, 218
_zebra_note_unnumbered:nnn	_zebra_zebra_label_name:n	
. 320, 326, 341 200, 204, 211, 222	
_zebra_pageref:n	_zebra_zebra_label_type:n	201, 223
199, 249, 252, 535	_zebra_zebra_ref:n	220, 230
_zebra_pdfstring_note:	\zebracomment	3, 4
_zebra_pdfstring_note_opt:w	\zebrafixed	4
. 194, 196	\zebranewnote	1, 4, 7,
_zebra_prepend:nn	19, 20, 22, 614, 618, 619, 620, 621, 622	
284, 500, 533	\zebranote	3
_zebra_print_note_group:n	\zebraplaceholder	4
545, 608	\zebraquestion	4
_zebra_print_notes:	\zebraref	2, 5, 22, 226, 236
574, 629	\zebrasetup	5, 7, 22, 616, 646
_zebra_print_notes_inorder:	\zebratodo	3
. 553, 605		
_zebra_record_note:nnnnn		
142, 303		
_zebra_render_note:nnn		
. 332, 367, 445, 456		
_zebra_saved_dbend:		
184, 187		
_zebra_setup_hyperref:		
245, 259		

Zebra Notes

Type	Count
todo	4
fixed	1
comment	1
note	3
placeholder	1
question	1
Total	11

List of notes

Todo 1 @alice	3
add two or three citations from the latest survey	
Todo 2 @bob, carol	3
verify the hyperparameter table against the source code and update any outdated entries and let's check afterwards	

Note 1	4
how should we structure the intro?	
Comment 1 @tom	4
the setup description looks clear now	
Placeholder 1 @lucy, tom	4
good job!	
Todo 3	4
check bibliography entries for formatting	
Fixed 1 @John	4
updated the conclusion	
Note 2 @who	4
anything else?	
Question 1 @who	5
what's this?	
Todo 4	5
revise the introduction before submission	
Note 3	5
see Todo 4 (that is, Todo 4) on p. 5	