WSKtracking — Tracking changes.

Wilfrid S. Kendall*

This version May 22, 2019

Tracking changes

```
Version 1.xx
 (...)
================================================
commit ...
Author: ...
Template for version file.
```

Contents

*E-mail: w.s.kendall@warwick.ac.uk

This package establishes some useful commands for tracking changes. They are especially useful when working collaboratively.

1 Packages

The presence of the following useful packages is ensured by the `WSKtracking` package:

- `graphicx`,
- `fancyvrb`,
- `multicol`,
- `ulem`,
- `framed`,
- `tocloft`,
- `xcolor` with options `[usenames,dvipsnames]`.

2 Preamble and Postamble

The `TrackingPreamble` and `TrackingPostamble` contain useful information about changes, comments, and versions.

`\TrackingPreamble` `TrackingPreamble` occurs at the start of this document. Note the need to pre-define `\Title` (to hold the document title!). Information provided includes:

- Title;
- Listing of a text file `VERSION.txt`, which should be placed in the same directory as the LaTeX source. Contents of `VERSION.txt` are generally at the discretion of the user. The example below is formatted as:
  1. Version number and date in first line,
  2. Second line is a separator,
  3. Third line contains the `git` hash for the latest commit;
  4. Fourth line describes the author;
  5. Fifth line left blank;
  6. Sixth line gives a brief description of most recent change.
- Table of contents;
- List of issues at this revision (both open issues and closed issues). item The optional argument is placed in a footnote to the title, set in fixed-width font.

`\listissuesname` Issues can be placed in the list of issues in the `TrackingPreamble` (whose title can be adjusted – it is produced by the macro `\listissuesname`). If `hyperref` is loaded then the lists are hyperlinked to the relevant pages. In practice `TrackingPreamble` is better placed *before* `\maketitle`.

`\TrackingPostamble` `TrackingPostamble` is best placed at the end of the document. It repeats a listing of the text file named `VERSION.txt`, and also lists the file given by the optional argument, if present.

```
Version 1.xx
 (...)
================================================
commit ...
Author: ...
Template for version file.
```

3 Displays to be completed and notes about incidental issues

\TBC The \TBC macro displays a list of matters needing attention:

```
\TBC{List matters needing attention here}
```

To Do: *List matters needing attention here*

An optional argument *arg* replaces **To Do:** by ***arg:*** as here:

```
\TBC[Attention]{List matters needing attention here}
```

Attention: *List matters needing attention here*

\NoteThis The \NoteThis macro displays a set of notes concerning detailed arguments which one may not wish to include in the final version of the paper.

```
\NoteThis{Notes go here}
```

To be omitted from public version.
Notes go here

If the option no-comment is given,

```
\includepackage[no-comment]{WSKtracking}
```

then all such notes are omitted.

4 Issues

\issue
\nonissue

Use `\issue{This needs attention}` to signal an issue that must not be forgotten.[1] Close an issue by using `\nonissue{No longer needs attention}` (hence, a simple edit of `\issue`).[2] It is recommended that each issue / nonissue be placed on its own line, to make it easier to remove when the time comes to produce the final version.

Normally one would use `\NB` and `\xNB` rather than `\issue` and `\nonissue`.

\issuesection

Larger issues can be placed in their own box, and the reference in the list of issues at the start then includes a note of the section concerned:

```
\issuesection{A sectional issue that is open.}
```

To Do: *A sectional issue that is open.*

\nonissuesection

It is easy to close such issues:

```
\nonissuesection{A sectional issue that is no longer open.}
```

If the option `no-comment` is given,

```
\includepackage[no-comment]{WSKtracking}
```

then all such issues (open or closed) are omitted.

5 Incidental Notes

\NB
\xNB

Incidental notes on issues can be attached using `\NB{First note}` thus[3], and can have two arguments thus `\NB[WSK1]{Second note}`, in which case first argument indicates person responsible for the note.[4] The issues can be closed easily thus `\xNB{First closed note}` or thus `\xNB[WSK1]{Second closed note}`, as can be seen.[5] [6] It is recommended that each note / closed note be placed on its own

[1] This needs attention
[2] ✓ ~~This no longer needs attention~~
[3] **Open:** First note
[4] **WSK1:** Second note
[5] ✓ ~~**Done:** First closed note~~
[6] ✓ ~~**WSK1:** Second closed note~~

line, to make it easier to remove when the time comes to produce the final version. Separate consecutive notes by `\!` to space out the footnote marks.

If the option `no-comment` is given,

```
\includepackage[no-comment]{WSKtracking}
```

then all such incidental notes (open or closed) are omitted.

6 Version control

Over the course of drafting and re-drafting a paper, it is common for authors to accumulate a confusing vaariety of different versions (`paper.tex`, `paper1.tex`, `paper-submitted.tex`, `paper-revised.tex`, ...) This is hard to control, and risks losing valuable edits in the forest of different versions!

Far better to learn how to use a good version-control system such as `rcs` (old-fashioned but still effective), `Subversion` (better for collaboration), or my current favourite `git` (very good, and simple useage is very simple). Version-control systems can store all previous versions efficiently (for example in compressed difference-based form) and invisibly (in hidden directories). Previous versions can be recovered at will (typically when exercising a little care to get the right previous version, and to avoid over-writing the current version: however in practice it is rare to need to recover a previous version other than the most recent one).

A typical workflow for `git` is as follows:

- `git init` establishes the `git` repository;
- `git add paper.tex paper.bib paper.pdf` places these files under `git` version control – note that they do not have to be text files!
- `git commit -a` updates the repository with the latest versions of all controlled files, and prompts you to add a comment describing this revision (one should get into the invariable habit of doing this after every revision no matter how small!). Just before this stage, it is useful to update the file `VERSION.txt` described above. (It is possible for `git`-adepts to make `git` do this automatically ....)
- `git checkout paper.tex` recovers `paper.tex` – be careful to rename current version temporarily if you don't wish it to be over-written!

See `http://git-scm.com/` for many more details and ideas for using `git`. A good book to consult is Lynn (2007), which provides a friendly introduction.

References

Ben Lynn. *Git Magic*. 2007. URL `http://www-cs-students.stanford.edu/~blynn/gitmagic/`.

7 Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.