

The pdfreview package

Michael Palmer

v1.1 (September 22, 2017)

Abstract

The `pdfreview` package lets you add comments in the page margins of PDF files, e.g. when reviewing manuscripts or grading reports. The PDF file to be annotated is included, one page at a time, as graphics, in a manner similar to the `pdfpages` package. Notes are placed in the margin next to the included graphics using a grid of help lines. Alternatively, only numbers are placed in the page margins, and the notes are collected into a numbered list at the end of the document.

Note that this package is *not* intended for adding notes directly to the L^AT_EX source of the document that is being reviewed; instead, the document undergoing review is already in PDF format and remains unchanged. Also note that this package does not produce the usual PDF ‘sticky notes’ that must be opened by clicking on them; instead, the notes are simply shown as text.

1 An example

Let’s assume a student submitted his essay as `lorem-ipsu.pdf`. In order to review and annotate it, I create this wrapper L^AT_EX document:

```
\documentclass[letterpaper,10pt]{article}

\usepackage[
  sourcedoc=lorem-ipsu.pdf,      % declare the PDF document being reviewed
  grid=true,                    % draw help lines for margin notes (set false when done)
  bodywidth=0.66,               % set width of source doc page as fraction of \textwidth
  twocolumn=true,               % place margin notes on both sides
  trim=2cm,                     % trim all page margins of source doc by this amount
]{pdfreview}

\begin{document}

\begin{page}{1}                 % mount the first page

\begin{leftnotes}               % write notes into the left margin
\cnote{84}{It looks like Latin.} % place note centered on help line 84
\bnote{68}{Is this really Latin?} % align bottom of note to help line 68
\tnote{16}{No, not Latin}        % align top of note to help line 16
\end{leftnotes}                 % notes after this point go into the right margin

\tnote{48}{You really should be writing proper Latin}
\cnote{12}{Cite your references}
\end{page}

% \begin{page}{2}                % mount the next page
% \note{...}{...} etc.

\end{document}
```

This gives us the following output (only the first page is shown):

| | | | |
|-----|-----------------------|--|-----|
| 100 | | | 100 |
| 96 | | | 96 |
| 92 | | | 92 |
| 88 | | | 88 |
| 84 | It looks like Latin. | | 84 |
| 80 | | | 80 |
| 76 | | | 76 |
| 72 | | | 72 |
| 68 | Is this really Latin? | | 68 |
| 64 | | | 64 |
| 60 | | | 60 |
| 56 | | | 56 |
| 52 | | | 52 |
| 48 | | | 48 |
| 44 | | | 44 |
| 40 | | | 40 |
| 36 | | | 36 |
| 32 | | | 32 |
| 28 | | | 28 |
| 24 | | | 24 |
| 20 | | | 20 |
| 16 | No, not Latin | | 16 |
| 12 | | | 12 |
| 8 | | | 8 |
| 4 | | | 4 |
| 0 | | | 0 |

When you have added all your comments, you can set the package option `grid=false` in order to hide the help lines. That is essentially it, already; some more options and a few tips and tricks will be given below.

2 Package options

sourcedoc (*<file name>*; default: `noname`). Name of the PDF file to be reviewed. The file extension (`.pdf`) can be omitted.

withnotesonly (*<boolean>*; default: `false`). By default, `pdfreview` will output every page that was mounted with `\begin{page} ... \end{page}`, even if no notes were placed on the page. If set to true, pages on which no notes were made will be omitted from the output. This can be useful to reduce file size.

Pages inserted with `\begin{insertpage} ... \end{insertpage}` will always be included in the output, regardless of whether or not they are empty.

twocolumn (*<boolean>*; default: **false**). If true, place included page in the middle and arrange for margin notes on both sides. If false, place included page on the left and have margin notes on the right hand side only.

inline (*<boolean>*; default: **true**). If true, print the note text into the page margin. If false, print only the number of each note into the page margin. In the latter case, the note text will be collected and printed out as an **enumerate**'d list at the end of the document. Numbers in the margin and list items will be reciprocally hyperlinked.

notenumbbers (*<boolean>*; default: **false**). If true, start each note with its number. If option **inline** is false, the number is printed regardless of this option.

grid (*<boolean>*; default: **true**). Draw help lines for margin notes.

gridcolor (*<color name>*; default: **black!30**). Color to use for help lines and line numbers.

maxscale (*<number>*; default: 100). Number of vertical length units/help lines for placing margin notes. If not divisible by 4, you will get a few ugly surplus lines at the top.

stretch (*<number>*; default: 1). By default, the vertical scale for placing margin notes is exactly as high as the first source document page. If that doesn't work well for some reason—for example, because you manually trim some pages more than others (see below)—then you can use this fudge factor to adjust the height of the scale manually. The scale will always be the same for all pages though.

bodywidth (*<number>*; default: 0.75). Width of the included source doc page, as fraction of `\textwidth` of the wrapper document. The remainder of `\textwidth` will be used for the margin notes.

The default of 0.75 should be suitable in combination with **inline=true,twocolumn=false**, that is, with full note text in the right margin only (which is my own most common use case). Use smaller values with **inline=true,twocolumn=true** or larger ones with **inline=false**.

trim (*<length(s)>*; default: 1cm). How much to trim off the margins of each page of the source document. This option is passed to the `\adjustbox` macro from the package by the same name. Accordingly, you can use two lengths to specify separate trimmings for horizontal and vertical margins, and four lengths to specify separate values for the left, lower, right, and top margins (in this order. No, it doesn't resemble a clock, fuel gauge, or anything else that mere mortals might have encountered).

Note that multiple lengths values must be surrounded with braces, e.g. **trim={1cm 2cm}** or **trim={1cm 2cm 0cm 1cm}**. If you omit the braces, only the first length value will be used, and the others will be silently ignored.

alignnotes (*<t, b, or c>*; default: **c**). Vertical alignment of a margin note declared with `\note`. Can be overridden by passing **t** or **b** as an optional argument (e.g. `\note[b]{Go boil your bottoms}` or `\note[t]{I shall taunt you a second time}`).

notesbg (*<color name or 'none'>*; default: **yellow**). Background color to use for the margin notes. If 'none', the background will be transparent.

notesframe (*<color name or 'none'>*; default: **black**). Color to use for the box frame of the margin notes. If 'none', no box is drawn.

notesep (*<length>*; default: 3pt). Empty space between included page and margin notes.

fontsize (*<string>*; default: **footnotesize**). Font size to use for the margin notes. Use any of the usual font size commands, but without the backslash.

pageoffset (*<number>*; default: 0). This option only affects the PDF bookmarks for each page of the current document; it is intended to synchronize the bookmarks with the page numbers of the source document.

By default, `pdfreview` simply numbers all included pages consecutively, starting with 1. If the source document has, say, 5 unnumbered (or Roman-numbered) pages preceding the proper page 1, you can declare this with `pageoffset=-5`; in this case, the first 5 pages will be given Roman numerals, and Arabic numbers will start (at 1) on the sixth physical page. Similarly, if the source document is missing some logical pages at the beginning, you can correct for this using a positive value for `pageoffset`.

(As usual, the least useful options require the most explanation.)

`insertpagemargin` (*<length>*; default: 2.5cm). If you have comments of a general nature, or ones that don't fit comfortably into the page margin, you can put them on separate pages insert one or more empty pages using the `insertpage` environment (see below).

By default, `pdfreview` leaves only very narrow empty page margins beyond the margin notes. For inserted text pages, it is more appropriate to use a wider margin; you can adjust it with this option.

3 Macros

`\note[alignment]{height}{text}` The optional `alignment` argument is one of `b`, `c`, or `t`. The *<height>* argument names the help line to align to; *<text>* is the content of the note. (The *<height>* argument doesn't have to be an integral number. Also, it can fall outside of the displayed help line scale, within reason.

Note that the entire text of each note will go inside a `TikZ` node. This means you cannot use empty lines within a note to insert paragraphs. If you do need paragraphs, use the `\par` macro instead.

`\bnote{height}{text}` Shorthand for `\note[b]{height}{text}`. Analogous `\cnote` and `\tnote`.

`\sourcedoc{file name}` Set the file name of the PDF file whose pages are to be included. More commonly, this is done using the `sourcedoc` package option (see above). However, this macro can be used to switch to another source document halfway through.

4 Environments

`\begin{page}[graphics options]{page number} ... \end{page}` This environment loads the physical page *<page number>* from the source document and sets up the container for the margin notes. The formatting and scaling of the included page and of the margin notes is subject to the package options described earlier.

The optional *<graphics options>* are passed to the `\adjustbox` macro from the package of the same name. You could use this for example to override the global `trim` package option for this one page, like so: `trim={0cm 1cm}`.

`\begin{leftnotes} ... \end{leftnotes}` This environment goes inside a `page` environment. In `twocolumn` mode, it will move the enclosed `\note` macros to the left page margin; with `twocolumn=false` it does nothing, that is, notes enclosed in `leftnotes` will show up in the right margin.

`\begin{rightnotes} ... \end{rightnotes}` Never does anything, ever; it only exists to avoid tripping up users who surmise its existence from that of the `leftnotes` environment.

`\begin{insertpage}[heading] ... \end{insertpage}` Inserts one or more extra pages for longer comments. These are not subject to any special formatting, except for the page margin, which can be set using a package option (see above). Alternatively, you can set the margins manually using `\newgeometry` and `\restoregeometry` from the `geometry` package, which is loaded by `pdfreview`. (The optional argument *<heading>* defaults to "General comments".)

5 Practical recommendations

5.1 Page space usage

Obviously, one needs to make some judicious choices in order to allow enough space for our margin comments yet also keep the included PDF pages legible. Use the `trim` package option to shave off empty margins from the included PDF pages.

The `geometry` package will be loaded to divide up the page. By default, `pdfreview` leaves 0.25 cm empty margins on both sides. If you load the `geometry` package yourself, `pdfreview` will respect your settings and not impose its own margin width. In that case, note that the entire content—including PDF page and margin notes—will go inside `\textwidth`. Any margins you declare using the `geometry` package will remain entirely empty. Also note that any `geometry` settings can be declared in the preamble only; if you try to change the margins within the document using `\newgeometry`, things will unravel.

5.2 Vertical overlap of margin notes

There is no automatic mechanism that would prevent mutual overlap of margin notes; this has to be prevented manually. Where space is tight, it may help to use e.g. a `\bnote{60}{some note}` followed by a `\tnote{60}{another note}`; no space is wasted between both notes, yet at the same time they will not overlap.

The `bodywidth` package option also decides the horizontal and vertical space available for the margin notes. Should you decide half-way through to reduce the `bodywidth` parameter, this will create more horizontal space for the notes, but at the same time reduce the vertical space. This may occasionally create some new vertical overlap that would have to be fixed manually.

Another consideration is what font to use for your notes, and at what size. Times is a nice, space-saving font (in pdfLaTeX, you can load it with `\usepackage{mathptmx}`). You can adjust the font size using the corresponding package option (e.g. `fontsize=scriptsize`).

5.3 Writing less boiler plate code

While the code to set up each page is pretty compact, it can still become tiresome to write for long source documents. The package includes a `bash` script named `pdfreview.sh`, which generates a document skeleton containing a `page` environment for each page of a source document:

```
mpalmer@holzkopf:/data$ pdfreview.sh source.pdf > review.tex
```

The script requires 1) `ghostscript`, 2) `bash` (obviously) and 3) that you place it somewhere into your shell's path.

5.4 File size

Particularly when working with large source files, `pdfreview` can produce some startlingly large PDF output files. This problem is apparently related to the way `\includegraphics` works and has been described before (see <https://tex.stackexchange.com/questions/207527/>). It can be amended by post-processing the output PDF file with `ghostscript`, for example:

```
gs -sDEVICE=pdfwrite -dCompatibilityLevel=1.4 -dPDFSETTINGS=/printer -dNOPAUSE -dQUIET \
  -dBATCH -sOutputFile=post-processed.pdf output.pdf
```

The package contains a `bash` script named `pdfshrink.sh` that wraps the call to `ghostscript` and which you can use like

```
mpalmer@holzkopf:/data$ pdfshrink.sh large.pdf
```

An optional second argument controls the resolution of included bitmap graphics, with possible values (in order of decreasing resolution) `prepress`, `printer`, `default` (the default value), `ebook`, and `screen`. A optional third argument can be used to indicate the output file name.

Again, you need to save the script in a directory listed in your shell's search path for this work.

5.5 Uncooperative input files

Occasionally, I've run into source documents that failed to cooperate with the page-wise inclusion attempted by `pdfreview`. I got around that by first printing the recalcitrant PDF input file to another PDF file—or, alternatively, running through `pdfshrink`—and then using the resulting file as the source document instead. (This might also be useful on occasion when using the `pdfpages` package.)

6 Implementation

This package depends on several others that are all in TeXLive. Among these is the `ulem` package, which makes the `\sout{strike me out}` command available for a strike-through effect. The `geometry` package is used for dividing up the page; if you load it yourself, your settings will be respected, as long you declare them in the preamble (but don't use `\newgeometry` halfway through the document). The `adjustbox` package is used to scale and trim the included pages, and also to scale the containers for the margin notes. The `environ` package is used to implement the `page` environment. The `tikz` package is used for placing help lines and margin notes.

For further details, see the package file itself.

7 Revision history

May 2017 initial version (1.00)

September 2017 corrections to documentation; added `withnotesonly` option (1.1)

8 License

All files in this package are subject to the LaTeX project public license (see <http://www.latex-project.org/lppl.txt>).