Creating a PDF document using PDFLaTeX

Nicola L C Talbot

17 July 2004

The home page for this document is http://www.dickimaw-books.com/latex/pdfdoc/. You can download a sample file illustrating the commands described in this document from the sample webpage.

Contents

1 Introduction

This is intended as a brief introduction to using PDFLaTeX. For more details, I recommend that you read "The LaTeX Web Companion" [1], and also the documentation for the hyperref package and the documentation for PDFTeX. You can use PDFLaTeX simply by using the command `pdflatex` instead of `latex`. For example if your document is called `filename.tex`, then instead of typing:

```
latex filename.tex
```

you would need to type:

```
pdflatex filename.tex
```

If you are using `TeXnicCenter` select the output profile `LaTeX => PDF`, and click on the 'Build' icon. If you are using `WinEdt`, click on the 'PDFLATEX' icon. If you are using some other front-end, check the manual.

2 Document Information

When you view a PDF document in `Acrobat Reader`, you can get the document information by selecting

<p align="center">File→Document Properties→Summary</p>

Figure 1 shows an example.

This information can be saved to the PDF file using the command:

$\pdfinfo\{info\}$

where *info* should be entered in PDF notation. For example:

```
\pdfinfo{
   /Author (Nicola Talbot)
   /Title  (Creating a PDF document using PDFLaTeX)
   /CreationDate (D:20040502195600)
   /Subject (PDFLaTeX)
   /Keywords (PDF;LaTeX)
}
```

If the creation date field is omitted, the current date and time is inserted. Note that all fields should be entered in the form:

/field name {text}

The date must be entered in the form: `D:YYYYMMDDHHmmss`. Available fields are:

```
/Title
/Author
/Creator
/Producer
/CreationDate
/ModDate
/Subject
/Keywords
```
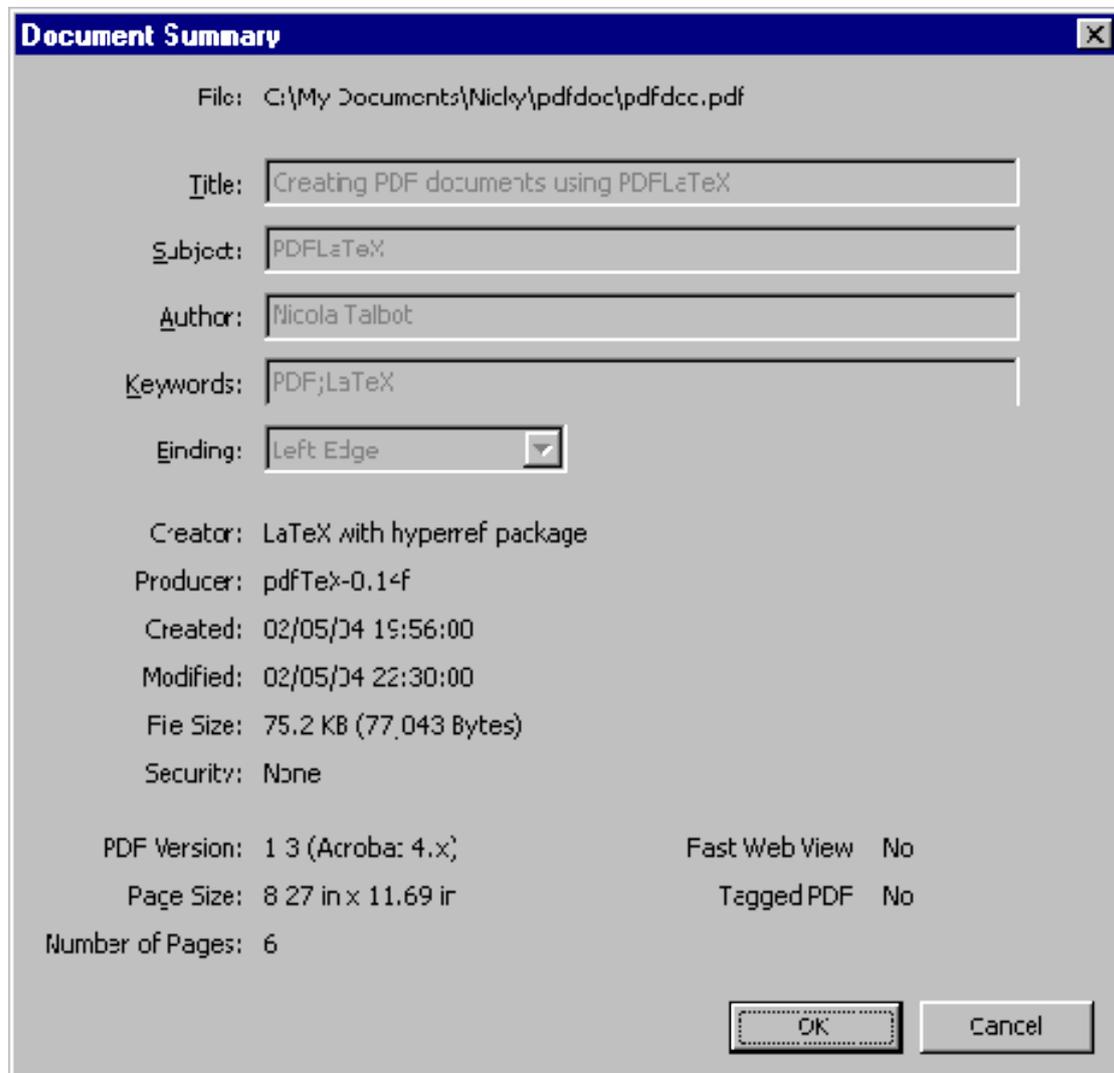
Figure 1: Document Properties

The field `/ModDate` indicates the modification date, and as with the creation date, the date should be entered in the form `D:YYYYMMDDHHmmss`. The `datetime` package (version 2.31 and above) has the command `\pdfdate` which can be used to insert the current date in the correct format. For example:

```
\pdfinfo{
   /Author (Nicola Talbot)
   /Title  (Creating a PDF document using PDFLaTeX)
   /CreationDate (D:20040502195600)
   /ModDate (D:\pdfdate)
   /Subject (PDFLaTeX)
   /Keywords (PDF;LaTeX)
}
```

Note that the command `\pdfinfo` is defined by PDFLaTeX[1] but not LaTeX, which means you'll get an error message if you try to use LaTeX instead of PDFLaTeX. The package `ifpdf` defines the conditional `\ifpdf` which can be used to determine whether you are using PDFLaTeX or LaTeX. For example the following code:

```
This is
\ifpdf
a PDF
\else
not a PDF
\fi
document.
```

will produce the output:

This is a PDF document.

if PDFLaTeX is used, otherwise it will produce the output:

This is not a PDF document.

So any commands that are specific to PDFLATEX (such as `\pdfinfo`) should be placed within `\ifpdf` ... `\fi`. For example:

```
\ifpdf
\pdfinfo{
   /Author (Nicola Talbot)
   /Title  (Creating PDF documents using PDFLaTeX)
   /CreationDate (D:20040502195600)
   /ModDate (D:\pdfdate)
```

---

[1] in fact, it's actually defined by PDFTeX

```
    /Subject (PDFLaTeX)
    /Keywords (PDF;LaTeX)
}
\fi
```

Note that if you are using the `ifthen` package, you can use

```
\ifthenelse{\boolean{pdf}}{...}{...}
```

instead of

```
\ifpdf...\else...\fi
```

(but you will still need the `ifpdf` package). For example, the following code:

```
This is
\ifthenelse{\boolean{pdf}}{a PDF}{not a PDF}
document.
```

will produce the output:

This is a PDF document.

if PDFLATEX is used, otherwise it will produce the output:

This is not a PDF document.

3 Including Graphics

As with LATEX, the `graphicx` (or `graphics`) package can be used with PDFLATEX, how-
ever you will no longer be able to include PostScript or Encapsulated PostScript images.
Instead, you can use PDF images (as well as a few other formats, such as PNG). There
are applications available for converting between various graphics formats, for example
`ps2pdf` and `eps2pdf`. If you want to have both a DVI and a PDF version of your docu-
ment, you would need to include the PostScript version of your image if using LATEX, and
the PDF version if you are using PDFLATEX. Suppose you have a file called `shapes.ps`
and you also have a PDF version called `shapes.pdf`, you could do:

```
\ifpdf
\includegraphics{shapes.pdf}
\else
\includegraphics{shapes.ps}
\fi
```

however it is simpler to omit the file extension:

```
\includegraphics{shapes}
```

If you are using PDFLATEX, the `graphicx` package will assume a `.pdf` or `.png` extension, otherwise it will assume a `.ps` or `.eps` extension.

If you like using `pstricks`, it is still possible to do so using PDFLATEX, however you will need to use the `pdftricks` package. Check the `pdftricks` documentation for further information.

4 The hyperref Package

We have already seen in section 2 that PDFTEX defines the command `\pdfinfo`. There are other commands that are also defined specifically for PDF documents, however as with all TEX commands, these commands are low-level. Fortunately, the `hyperref` package provides an easy interface to these commands. If you want to use the `hyperref` package it is recommended that you read the `hyperref` documentation and also Chapter 2 of "The LATEX Web Companion" [1], as this document merely gives a brief overview of the available options. Options can either be specified as a comma-separated list of key=value pairs in the optional argument to the `hyperref` package, e.g.

```
\usepackage[pdfpagemode=FullScreen,bookmarks=true]{hyperref}
```

or as the argument to the command `\hypersetup`, e.g.

```
\hypersetup{pdftoolbar=false}
```

If you are switching on an option, you can omit =true, e.g. `\hypersetup`{bookmarks}. One more thing to note: the `hyperref` package must always be the last package to be included (unless of course, you are using a package that modifies the behaviour of the hyperref package, e.g. `backrefx`).

4.1 Cross-References and Citations

All the cross-references and citations (using `\ref`, `\pageref` and `\cite`) will automatically be converted into active links in your document when you use the `hyperref` package. The default action is to place hyperlinks in a rectangle. For example:

```
See section~\ref{sec:hyperref}
```

will by default look like:

See section 4

You can instead choose to omit the box and simply colour the text by selecting the `colorlinks` option. (e.g. `\usepackage[colorlinks]{hyperref}`). The above example would then look like:

See section 4

Alternatively, the command \autoref can be used which will insert the correct context name in front of the number. For example:

```
See \autoref{sec:hyperref}
```

will look like:

See section 4

and

```
See \autoref{fig:docinfo}
```

will look like:

See Figure 1

The context name is determined as follows: firstly, the \label command is redefined by the hyperref package so that the name of the counter to which it's referring is stored in the auxiliary file. For example, Figure 1 contained the following code:

```
\caption{Document Properties}
\label{fig:docinfo}
```

This produces the following entry in the auxiliary file:

```
\newlabel{fig:docinfo}{{1}{2}{Document Information\relax }{figure.1}{}}
```

In this case, the relevant counter is figure, so the \autoref command will use the command \figurename, if it exists, to generate the context name. So, if you want to define a new counter that you want to reference using \autoref, you will also need to define the corresponding *counter-name*name command. For example:

```
\newcounter{exercise}
\newcommand{\exercisename}{Exercise}
```

Another way of creating a hyperlink is to use the command

\hyperref[*label*]{*text*}

For example, the following code:

```
In the \hyperref[sec:intro]{introduction} \ldots
```

would produce the following output:

In the introduction ...

The hyperref package also provides starred versions of the commands \ref and \pageref which print the relevant number, but do not create a hyperlink. These can be used within the \hyperref command. For example:

```
\hyperref[sec:hyperref]{See section~\ref*{sec:hyperref}
(on page~\pageref*{sec:hyperref})}
```

would look like:

See section 4 (on page 6)

Hyperlinks to URLs can be created using the command:

$\href\{url\}\{text\}$

For example:

```
See the
\href{http://theoval.cmp.uea.ac.uk/~nlct/latex/csed/}{course web site}
for further details.
```

This would produce the following:

See the course web site for further details.

Note that you don't need to worry about the tilde in the first argument to \href, nor do you have to worry about escaping the # character:

```
\href{http://theoval.cmp.uea.ac.uk/~nlct/index.html#latex}%
{\LaTeX\ information}
```

This would produce:

LaTeX information

Alternatively, if you simply want to print the web address as an active link, you can use:

$\url\{address\}$

For example:

```
\url{http://theoval.cmp.uea.ac.uk/~nlct}
```

would produce:

http://theoval.cmp.uea.ac.uk/~nlct

The hyperref package also turns \cite commands into active links.

```
See ``The \LaTeX\ Web Companion''~\cite[Chapter~2]{goossens1999}
for further details.
```

This would produce:

See "The LaTeX Web Companion" [1, Chapter 2] for further details.

The `backref` package can be used to create a set of back-references within the bibliography. To implement this, use the `backref` option to the hyperref package. By default this will reference the section number, but you can change it to reference the page instead, by doing either

```
\usepackage[backref=page]{hyperref}
```

or

```
\usepackage[pagebackref]{hyperref}
```

There is another package `backrefx` which extends the `backref` package, so that the back-references are in the form: (Cited on pages 1, 4 and 6.) instead of a simple list of numbers. The `backrefx` package should be included after the `hyperref` package, e.g.:

```
\usepackage[pagebackref]{hyperref}
\usepackage{backrefx}
```

Note that both the `backref` and `backrefx` packages assume that the bibliographic entries are separated by a paragraph break. This is done automatically by BibTeX, but it is something that you will need to remember if you are writing the `thebibliography` environment by yourself.

4.2 Table of Contents and Bookmarks

The table of contents (produced as usual with the `\tableofcontents` command) will automatically have each entry as a hyperlink. By default, the headings rather than the page number will be the hyperlink to the relevant chapter etc. The option `linktocpage` will swap this round. If the option `bookmarks` is set, a set of PDF bookmarks will be created, allowing you to navigate your way around the document. The bookmarks can be viewed in `Acrobat Reader` either by clicking on the bookmarks tab, or selecting the menu option

$$\text{Window} \rightarrow \text{Bookmarks}$$

The option `bookmarksopen` can be set if you want all the bookmark subtrees expanded, and the option `bookmarksnumbered` can be set if you want the section numbers included in the bookmarks. The bookmark entries are taken from the chapter, section etc headings, but note that the text in the PDF bookmark will not follow any LaTeX formatting.

4.2.1 An Example Heading $a^2 + b^2 = c^2$

This heading was produced with the following code:

```
\subsubsection{An Example Heading $a^2+b^2=c^2$}
```

If you view the PDF version of this document in `Acrobat Reader`, you will be able to see that this section heading in the bookmarks has appeared as: An Example Heading: a2+b2=c2. LaTeXing this document produces the following warnings:

```
Package hyperref Warning: Token not allowed in a PDFDocEncoded string,
(hyperref)                removing 'math shift' on input line 388.
```

```
Package hyperref Warning: Token not allowed in a PDFDocEncoded string,
(hyperref)                removing 'superscript' on input line 388.
```

This means that the `$` and `^` symbols have been ignored. The command

```
\texorpdfstring{LaTeX text}{PDF text}
```

can be used to vary the text, depending on whether it is to be processed by LATEX or whether it will appear in the PDF bookmark. The above section heading could then be changed to

```
\subsubsection{An Example Heading\texorpdfstring{: $a^2+b^2=c^2$}{}}
```

In this case, the equation will be printed in the section heading within the document, but will not appear in the bookmark.

4.2.2 Duplicate Page Numbers

Bookmark problems can occur when you have duplicate page numbers, for example, if you have a page i and a page 1. This leads to the warning:

```
! pdfTeX warning (ext4): destination with the same identifier
(name{page.1}) has been already used, duplicate ignored
```

This can be overcome by switching off the `plainpages` option:

```
\usepackage[plainpages=false]{hyperref}
```

This will change the identifiers so that they are constructed using the formatted form of the page number instead of solely the Arabic form (e.g. page.i instead of page.1) You may still find a problem arising from the title page using, say the report class, where although the page number does not appear, it is assigned the page number 1, but the first page of your main matter is also page 1. This problem can be overcome by using a different page numbering style for the title page, that does not occur in the rest of the document. For example:

```
\documentclass{report}

\usepackage[plainpages=false]{hyperref}

\begin{document}
\title{A Sample Document}
\author{Me}

\pagenumbering{alph}
\maketitle

\clearpage\pagenumbering{roman}
\tableofcontents

\clearpage\pagenumbering{arabic}
\chapter{Introduction}
```

In this case, switching to alph pagenumbering does not affect the visual formatting as the page number does not appear on the title page, but it will yield unique identifiers.

4.3 Page Layouts

The paper size can be set by passing one of the following options to the `hyperref` package:

Option	Size
a4paper	210mm x 297mm
a5paper	148mm x 210mm
b5paper	176mm x 250mm
letterpaper	8.5in x 11in
legalpaper	8.5in x 14in
executivepaper	7.25in x 10.5in

It is also possible to set the following options which can be used to change the display options (see "The LaTeX Web Companion" [1] for a complete list):

Option	Action
pdfcenterwindow	Position window displaying document in centre of the screen
pdffitwindow	Resizes window displaying document to fit the size of the first displayed page of the document.
pdftoolbar	Sets the toolbar to be visible (=true) or not visible (=false)
pdfmenubar	Sets the menu bar to be visible (=true) or not visible (=false)
pdfpagelayout	Sets the page layout when the document is opened. This can take the values: SinglePage, OneColumn, TwoColumnLeft, TwoColumnRight

4.4 Acrobat Menu Command

The command

\Acrobatmenu{*menuitem*}{*text*}

can be used to access the relevant `Acrobat` menu item, where *menuitem* is the Acrobat menu option name and *text* is the link text. For example:

```
\Acrobatmenu{GeneralInfo}{Document Summary}
```

will produce an active link containing the text

Document Summary

which will produce the general information dialogue box if you click on it when viewing the PDF document using Acrobat Reader. Note that text doesn't have to simply be text, it can be any LATEX code, which means you could use a picture instead. A full list of menu options is given in "The LaTeX Web Companion" [1], but the most commonly used ones are: `PrevPage`, `NextPage`, `FirstPage`, `LastPage`, `GoBack`, `GoForward` and `Quit`. It is therefore possible to incorporate your own navigation panel within your document by defining a new page style that includes \Acrobatmenu commands. For example, this document defines a new page style called online, and is defined as follows:

```
\newcommand\ps@online{
\renewcommand{\@oddhead}{}
\renewcommand{\@evenhead}{}
\renewcommand{\@oddfoot}{\hfill
   \Acrobatmenu{PrevPage}{Previous}
   \Acrobatmenu{NextPage}{Next}
   \Acrobatmenu{FirstPage}{First}
```

```
    \Acrobatmenu{LastPage}{Last}
    \Acrobatmenu{GoBack}{Back}
    \Acrobatmenu{GoForward}{Forward}
    \hyperref[sec:index]{Index}\hfill\thepage}
\renewcommand{\@evenfoot}{\@oddfoot}}
```

As can be seen the headers are blank, and the footer contains the \Acrobatmenu commands. It also uses the \hyperref command to provide a link to the index, and the \hfill commands neatly centre the text, with the page number pushed to the far right. The theindex environment was also redefined so that it incorporated the \label{sec:index} command. This was necessary because \printindex starts a new page, so placing the label before this command would link to the page before the index, and placing the label after \printindex would link to the end of the index.

5 Strange Errors or Unexpected Ouput

This section lists some errors or unexpected results that you may encounter using PDFLATEX.

- The links in my index and backref citations go to the absolute page rather than the LATEX page.

  There are some packages that seem to interfere with the commands that generate these links. I've noticed that using the subfigure package seems to cause this problem. (I haven't worked out why.)

- I get the error

```
pdfTeX error (ext4): link annotations can't be nested.
\grf@@shipout ->\grf@org@shipout \box \@cclv
                                          \relax
```

  but I don't have any nested annotations.

  This error can be caused by a page break occurring in the middle of a link. For example, suppose you have the following:

```
\hyperref[sec:optional]{optional arguments}
```

  If a page break occurs between the words "optional" and "arguments", a shipout command will occur whilst a link is still being created, which will cause the error. To get around this, you can split the link up into two links:

```
\hyperref[sec:optional]{optional}
\hyperref[sec:optional]{arguments}
```

- Some of my fonts haven't come out.

  PDFLATEX can't handle PostScript fonts, so check to see if you have included any package that use PostScript fonts (such as `pifont`).

- I'm using the `xr` package to cross-reference a label defined in another document, but my `\ref` command is producing something like:

  ```
  1Introductionsection.1
  ```

  instead of simply the number 1.

  This is caused by the fact that the document containing the label definition uses the `hyperref` package, and the document referencing doesn't. Recall from section 4.1 that the `hyperref` package redefines the `\label` command. The line in the auxiliary file now contains additional information, and since both documents read in the same auxiliary file, they must both have the same definitions of `\label`.

- I'm using the xr package to cross-reference a label defined in another document, but I get the error:

  ```
  ! Argument of \@fifthoffive has an extra }
  ```

  This is the same kind of problem as the previous one, but in this case the `\label` has been defined in a file that doesn't include the `hyperref` package, but the document referencing it does.

References

[1] Michel Goossens, Sebastian Rahtz, et al. *The LATEX web companion.* Addison-Wesley, 1999.

Index