Hypertext marks in LaTeX: a Manual for hyperref

Sebastian Rahtz Heiko Oberdiek

July 2003

Contents

1 Introduction

The package derives from, and builds on, the work of the HyperTeX project, described at `http://xxx.lanl.gov/hypertex/`. It extends the functionality of all the LaTeX cross-referencing commands (including the table of contents, bibliographies etc) to produce `\special` commands which a driver can turn into hypertext links; it also provides new commands to allow the user to write *ad hoc* hypertext links, including those to external documents and URLs.

This manual provides a brief overview of the hyperref package. For more details, you should read the additional documentation distributed with the package, as well as the complete documentation by processing `hyperref.dtx`. You should also read the chapter on hyperref in *The LaTeX Web Companion*, where you will find additional examples.

The HyperTEX specification[1] says that conformant viewers/translators must recognize the following set of `\special` constructs:

href: `html:<a href = "href_string">`

name: `html:<a name = "name_string">`

end: `html:</a>`

image: `html:<img src = "href_string">`

base_name: `html:<base href = "href_string">`

The *href*, *name* and *end* commands are used to do the basic hypertext operations of establishing links between sections of documents. The *image* command is intended (as with current HTML viewers) to place an image of arbitrary graphical format on the page in the current location. The *base_ name* command is be used to communicate to the DVI viewer the full (URL) location of the current document so that files specified by relative URL's may be retrieved correctly.

The *href* and *name* commands must be paired with an *end* command later in the TEX file—the TEX commands between the two ends of a pair form an *anchor* in the document. In the case of an *href* command, the *anchor* is to be highlighted in the *DVI viewer*, and when clicked on will cause the scene to shift to the destination specified by *href_ string*. The *anchor* associated with a name command represents a possible location to which other hypertext links may refer, either as local references (of the form `href="#name_string"` with the *name_ string* identical to the one in the name command) or as part of a URL (of the form *URL#name_ string*). Here *href_ string* is a valid URL or local identifier, while *name_ string* could be any string at all: the only caveat is that '"' characters should be escaped with a backslash (\), and if it looks like a URL name it may cause problems.

However, the drivers intended to produce *only* PDF use literal PostScript or PDF `\special` commands. The commands are defined in configuration files for different drivers, selected by package options; at present, the following drivers are supported:

hypertex DVI processors conforming to the HyperTEX guidelines (i.e. xdvi, dvips (with the -z option), OzTeX, and Textures)

dvips produces `\special` commands tailored for dvips

dvipsone produces `\special` commands tailored for dvipsone

ps2pdf a special case of output suitable for processing by earlier versions of Ghostscript's PDF writer; this is basically the same as that for dvips, but a few variations remained before version 5.21

tex4ht produces `\special` commands for use with TEX4ht

pdftex pdfTEX, Hàn Thế Thành's TEX variant that writes PDF directly

dvipdf produces `\special` commands for the DVI to PDF driver dvipdf

dvipdfm produces `\special` commands for Mark Wicks' DVI to PDF driver dvipdfm

dviwindo produces `\special` commands that Y&Y's Windows previewer interprets as hypertext jumps within the previewer

vtex produces `\special` commands that MicroPress' HTML and PDF-producing TEX variants interpret as hypertext jumps within the previewer

[1] This is borrowed from an article by Arthur Smith.

textures produces `\special` commands that Textures interprets as hypertext jumps within the previewer

Output from dvips or dvipsone must be processed using Acrobat Distiller to obtain a PDF file.[2] The result is generally preferable to that produced by using the hypertex driver, and then processing with dvips -z, but the DVI file is not portable. The main advantage of using the HyperTEX `\special` commands is that you can also use the document in hypertext DVI viewers, such as xdvi.

2 Implicit behavior

This package can be used with more or less any normal LATEX document by specifying in the document preamble

```
\usepackage{hyperref}
```

Make sure it comes *last* of your loaded packages, to give it a fighting chance of not being over-written, since its job is to redefine many LATEX commands. Hopefully you will find that all cross-references work correctly as hypertext. For example, `\section` commands will produce a bookmark and a link, whereas `\section*` commands will only show links when paired with a corresponding `\addcontentsline` command.

In addition, the `hyperindex` option (see below) attempts to make items in the index by hyperlinked back to the text, and the option `backref` inserts extra 'back' links into the bibliography for each entry. Other options control the appearance of links, and give extra control over PDF output. For example, `colorlinks`, as its name well implies, colors the links instead of using boxes; this is the option used in this document.

3 Package options

All user-configurable aspects of hyperref are set using a single 'key=value' scheme (using the keyval package) with the key `Hyp`. The options can be set either in the optional argument to the `\usepackage` command, or using the `\hypersetup` macro. When the package is loaded, a file `hyperref.cfg` is read if it can be found, and this is a convenient place to set options on a site-wide basis.

As an example, the behavior of a particular file could be controlled by:

- a site-wide `hyperref.cfg` setting up the look of links, adding backreferencing, and setting a PDF display default:

  ```
  \hypersetup{backref,
  pdfpagemode=FullScreen,
  colorlinks=true}
  ```

- A global option in the file, which is passed down to hyperref:

  ```
  \documentclass[dvips]{article}
  ```

- File-specific options in the `\usepackage` commands, which override the ones set in `hyperref.cfg`:

[2] Make sure you turn off the partial font downloading supported by dvips and dvipsone in favor of Distiller's own system.

```
\usepackage[pdftitle={A Perfect Day},colorlinks=false]{hyperref}
```

Some options can be given at any time, but many are restricted: before `\begin{document}`, only in `\usepackage[...]{hyperref}`, before first use, etc.

In the key descriptions that follow, many options do not need a value, as they default to the value true if used. These are the ones classed as 'boolean'. The values true and false can always be specified, however.

3.1 General options

Firstly, the options to specify general behavior and page size.

`draft`	boolean	*false*	all hypertext options are turned off
`final`	boolean	*true*	all hypertext options are turned on
`debug`	boolean	*false*	extra diagnostic messages are printed in the log file
`verbose`	boolean	*false*	same as debug
`implicit`	boolean	*true*	redefines LaTeX internals
`hypertexnames`	boolean	*true*	use guessable names for links
`naturalnames`	boolean	*false*	use LaTeX-computed names for links
`a4paper`	boolean	*true*	sets paper size to 210mm × 297mm
`a5paper`	boolean	*false*	sets paper size to 148mm × 210mm
`b5paper`	boolean	*false*	sets paper size to 176mm × 250mm
`letterpaper`	boolean	*false*	sets paper size to 8.5in × 11in
`legalpaper`	boolean	*false*	sets paper size to 8.5in × 14in
`executivepaper`	boolean	*false*	sets paper size to 7.25in × 10.5in
`setpagesize`	boolean	*true*	sets page size by special driver commands

3.2 Configuration options

`raiselinks`	boolean	*true*	In the hypertex driver, the height of links is normally calculcated by the driver as simply the base line of contained text; this options forces `\special` commands to reflect the real height of the link (which could contain a graphic)
`breaklinks`	boolean	*false*	Allows link text to break across lines; since this cannot be accommodated in PDF, it is only set true by default if the pdftex driver is used. This makes links on multiple lines into different PDF links to the same target.
`pageanchor`	boolean	*true*	Determines whether every page is given an implicit anchor at the top left corner. If this is turned off, `\tableofcontents` will not contain hyperlinks.
`plainpages`	boolean	*true*	Forces page anchors to be named by the arabic form of the page number, rather than the formatted form.
`nesting`	boolean	*false*	Allows links to be nested; no drivers currently support this.

3.3 Backend drivers

If no driver is specified, the package defaults to loading the hypertex driver. All of these are boolean options.

`dvips`	Sets up hyperref for use with the dvips driver.
`dvipsone`	Sets up hyperref for use with the dvipsone driver.
`dviwindo`	Sets up hyperref for use with the dviwindo Windows previewer.
`hypertex`	Sets up hyperref for use with the HyperTEX-compliant drivers.
`latex2html`	Redefines a few macros for compatibility with latex2html.
`nativepdf`	an alias for dvips
`pdfmark`	an alias for dvips
`pdftex`	Sets up hyperref for use with the pdftex program.
`ps2pdf`	Redefines a few macros for compatibility with Ghostscript's PDF writer, otherwise identical to dvips.
`tex4ht`	for use with TEX4ht
`textures`	for use with Textures
`vtex`	For use with MicroPress' VTeX; the PDF and HTML backends are detected automatically.
`vtexpdfmark`	for use with VTeX's PostScript backend.

If you use dviwindo, you may need to redefine the macro `\wwwbrowser` (the default is `C:\netscape\netscape`) to tell dviwindo what program to launch. Thus, users of Internet Explorer might add something like this to hyperref.cfg:

```
\renewcommand{wwwbrowser}{C:\string\Program\space
  Files\string\Plus!\string\Microsoft\space
  Internet\string\iexplore.exe}
```

3.4 Extension options

`extension`	text		Set the file extension (e.g. dvi) which will be appended to file links created if you use the xr package.
`hyperfigures`	boolean		
`backref`	boolean	*false*	Adds 'backlink' text to the end of each item in the bibliography, as a list of section numbers. This can only work properly *if* there is a blank line after each `\bibitem`.
`pagebackref`	boolean	*false*	Adds 'backlink' text to the end of each item in the bibliography, as a list of page numbers.
`hyperindex`	boolean	*false*	Makes the text of index entries into hyperlinks. Easily broken ...
`encap`			Sets encap character for hyperindex
`linktocpage`	boolean	*false*	make page number, not text, be link on TOC, LOF and LOT
`breaklinks`	boolean	*false*	allow links to break over lines by making links over multiple lines into PDF links to the same target

`colorlinks`	boolean	*false*	Colors the text of links and anchors. The colors chosen depend on the the type of link. At present the only types of link distinguished are citations, page references, URLs, local file references, and other links.
`linkcolor`	color	*red*	Color for normal internal links.
`anchorcolor`	color	*black*	Color for anchor text.
`citecolor`	color	*green*	Color for bibliographical citations in text.
`filecolor`	color	*magenta*	Color for URLs which open local files.
`menucolor`	color	*red*	Color for Acrobat menu items.
`pagecolor`	color	*red*	Color for links to other pages.
`urlcolor`	color	*cyan*	Color for linked URLs.
`frenchlinks`	boolean	*false*	use small caps instead of color for links

Note that all color names must be defined before use, following the normal system of the standard LaTeX color package.

3.5 PDF-specific display options

`bookmarks`	boolean	*false*	A set of Acrobat bookmarks are written, in a manner similar to the table of contents, requiring two passes of LaTeX. Some postprocessing of the bookmark file (file extension `.out`) may be needed to translate LaTeX codes, since bookmarks must be written in PDFEncoding. To aid this process, the `.out` file is not rewritten by LaTeX if it is edited to contain a line `\let\WriteBookmarks\relax`
`bookmarksopen`	boolean	*false*	If Acrobat bookmarks are requested, show them with all the subtrees expanded.
`bookmarksopenlevel`	parameter		level (`\maxdimen`) to which bookmarks are open
`bookmarksnumbered`	boolean	*false*	If Acrobat bookmarks are requested, include section numbers.
`bookmarkstype`	text	*toc*	to specify which 'toc' file to mimic
`pdfhighlight`	name	*/I*	How link buttons behave when selected; /I is for inverse (the default); the other possibilities are /N (no effect), /O (outline), and /P (inset highlighting).
`citebordercolor`	RGB color	*0 1 0*	The color of the box around citations

`filebordercolor`	RGB color	*0 .5 .5*	The color of the box around links to files
`linkbordercolor`	RGB color	*1 0 0*	The color of the box around normal links
`menubordercolor`	RGB color	*1 0 0*	The color of the box around Acrobat menu links
`pagebordercolor`	RGB color	*1 1 0*	The color of the box around links to pages
`urlbordercolor`	RGB color	*0 1 1*	The color of the box around links to URLs
`runbordercolor`	RGB color	*0 .7 .7*	color of border around 'run' links
`pdfborder`		*0 0 1*	The style of box around links; defaults to a box with lines of 1pt thickness, but the colorlinks option resets it to produce no border.

Note that the color of link borders can be specified *only* as 3 numbers in the range 0..1, giving an RGB color. You cannot use colors defined in TEX.

The bookmark commands are stored in a file called *jobname*`.out`. The files is not processed by LATEX so any markup is passed through. You can postprocess this file as needed; as an aid for this, the `.out` file is not overwritten on the next TEX run if it is edited to contain the line

```
\let\WriteBookmarks\relax
```

3.6 PDF display and information options

`baseurl`	URL		Sets the base URL of the PDF document
`pdfpagemode`	text	*None*	Determines how the file is opening in Acrobat; the possibilities are `None`, `UseThumbs` (show thumbnails), `UseOutlines` (show bookmarks), and `FullScreen`. If no mode if explicitly chosen, but the bookmarks option is set, `UseOutlines` is used.
`pdftitle`	text		Sets the document information Title field
`pdfauthor`	text		Sets the document information Author field
`pdfsubject`	text		Sets the document information Subject field
`pdfcreator`	text		Sets the document information Creator field
`pdfproducer`	text		Sets the document information Producer field
`pdfkeywords`	text		Sets the document information Keywords field
`pdfview`	text	*FitBH*	Sets the default PDF 'view' for each link

`pdfstartpage`	text	*1*	Determines on which page the PDF file is opened.
`pdfstartview`	text	*FitB*	Set the startup page view
`pdfpagescrop`	n n n n		Sets the default PDF crop box for pages. This should be a set of four numbers
`pdfcenterwindow`	boolean	*false*	position the document window in the center of the screen
`pdffitwindow`	boolean	*false*	resize document window to fit document size
`pdfmenubar`	boolean	*true*	make PDF viewer's menu bar visible
`pdfnewwindow`	boolean	*false*	make links that open another PDF file start a new window
`pdfpagelayout`	text	*empty*	set layout of PDF pages
`pdfpagelabels`	boolean	*false*	set PDF page labels
`pdfpagetransition`	text	*empty*	set PDF page transition style
`pdftoolbar`	boolean	*true*	make PDF toolbar visible
`pdfwindowui`	boolean	*true*	make PDF user interface elements visible
`unicode`			Unicode encoded PDF strings

Each link in Acrobat carries its own magnification level, which is set using PDF coordinate space, which is not the same as TEX's. pdfTEX works by supplying default values for `XYZ` (horizontal × vertical × zoom) and `FitBH`. However, drivers using `pdfmark` do not supply defaults, so hyperref passes in a value of -32768, which causes Acrobat to set (usually) sensible defaults. The following are possible values for the `pdfview` and `pdfstartview` parameters.

`XYZ`	*left top zoom*	Sets a coordinate and a zoom factor. If any one is null, the source link value is used. *null null null* will give the same values as the current page.
`Fit`		Fits the page to the window.
`FitH`	*top*	Fits the width of the page to the window.
`FitV`	*left*	Fits the height of the page to the window.
`FitR`	*left bottom right top*	Fits the rectangle specified by the four coordinates to the window.
`FitB`		Fits the page bounding box to the window.
`FitBH`	*top*	Fits the width of the page bounding box to the window.
`FitBV`	*left*	Fits the height of the page bounding box to the window.

The `pdfpagelayout` can be one of the following values.

`SinglePage`	Displays a single page; advancing flips the page
`OneColumn`	Displays the document in one column; continuous scrolling.
`TwoColumnLeft`	Displays the document in two columns, odd-numbered pages to the left.
`TwoColumnRight`	Displays the document in two columns, odd-numbered pages to the right.

Finally, the `pdfpagetransition` can be one of the following values, where */Di* stands for

direction of motion in degrees, generally in 90° steps, */Dm* is a horizontal (`/H`) or vertical (`/V`) dimension (e.g. `Blinds /Dm /V`), and */M* is for motion, either in (`/I`) or out (`/O`).

`Blinds`	/Dm	Multiple lines distributed evenly across the screen sweep in the same direction to reveal the new page.
`Box`	/M	A box sweeps in or out.
`Dissolve`		The page image dissolves in a piecemeal fashion to reveal the new page.
`Glitter`	/Di	Similar to Dissolve, except the effect sweeps across the screen.
`Split`	/Dm /M	Two lines sweep across the screen to reveal the new page.
`Wipe`	/Di	A single line sweeps across the screen to reveal the new page.

3.7 Big alphabetical list

The following is a complete listing of available options for hyperref, arranged alphabetically.

`a4paper`		use A4 paper
`a5paper`		use A5 paper
`anchorcolor`	*black*	set color of anchors
`b5paper`		use B5 paper
`backref`	*false*	do bibliographical back references
`baseurl`	*empty*	set base URL for document
`bookmarks`	*true*	make bookmarks
`bookmarksnumbered`	*false*	put section numbers in bookmarks
`bookmarksopen`	*false*	open up bookmark tree
`bookmarksopenlevel`	`\maxdimen`	level to which bookmarks are open
`bookmarkstype`	*toc*	to specify which 'toc' file to mimic
`breaklinks`	*false*	allow links to break over lines
`citebordercolor`	*0 1 0*	color of border around cites
`citecolor`	*green*	color of citation links
`colorlinks`	*false*	color links
	true	(tex4ht, dviwindo)
`debug`	*false*	provide details of anchors defined; same as verbose
`draft`	*false*	do not do any hyperlinking
`dvipdf`		use dvipdf backend
`dvipdfm`		use dvipdfm backend
`dvips`		use dvips backend
`dvipsone`		use dvipsone backend
`dviwindo`		use dviwindo backend
`encap`		to set encap character for hyperindex
`executivepaper`		use executivepaper
`extension`	*dvi*	suffix of linked files
`filebordercolor`	*0 .5 .5*	color of border around file links
`filecolor`	*cyan*	color of file links
`final`	*true*	opposite of option draft
`frenchlinks`	*false*	use small caps instead of color for links
`hyperfigures`	*false*	make figures hyper links
`hyperindex`	*true*	set up hyperlinked indices
`hypertex`		use HyperTEX backend

`hypertexnames`	*true*	use guessable names for links
`implicit`	*true*	redefine LaTeX internals
`latex2html`		use LaTeX2HTML backend
`legalpaper`		use legalpaper
`letterpaper`		use letterpaper
`linkbordercolor`	*1 0 0*	color of border around links
`linkcolor`	*red*	color of links
`linktocpage`	*false*	make page number, not text, be link on TOC, LOF and LOT
`menubordercolor`	*1 0 0*	color of border around menu links
`menucolor`	*red*	color for menu links
`nativepdf`	*false*	an alias for dvips
`naturalnames`	*false*	use LaTeX-computed names for links
`nesting`	*false*	allow nesting of links
`pageanchor`	*true*	put an anchor on every page
`pagebackref`	*false*	backreference by page number
`pagebordercolor`	*1 1 0*	color of border around page links
`pagecolor`	*red*	color of page links
`pdfauthor`	*empty*	text for PDF Author field
`pdfborder`	*0 0 1*	width of PDF link border
	0 0 0	(`colorlinks`)
`pdfcenterwindow`	*false*	position the document window in the center of the screen
`pdfcreator`	*LaTeX with hyperref package*	text for PDF Creator field
`pdffitwindow`	*false*	resize document window to fit document size
`pdfhighlight`	*/I*	set highlighting of PDF links
`pdfkeywords`	*empty*	text for PDF Keywords field
`pdfmark`	*false*	an alias for dvips
`pdfmenubar`	*true*	make PDF viewer's menu bar visible
`pdfnewwindow`	*false*	make links that open another PDF file start a new window
`pdfpagelayout`	*empty*	set layout of PDF pages
`pdfpagemode`	*empty*	set default mode of PDF display
`pdfpagelabels`	*false*	set PDF page labels
`pdfpagescrop`	*empty*	set crop size of PDF document
`pdfpagetransition`	*empty*	set PDF page transition style
`pdfproducer`	*empty*	text for PDF Producer field
`pdfstartpage`	*1*	page at which PDF document opens
`pdfstartview`	*/Fit*	starting view of PDF document
`pdfsubject`	*empty*	text for PDF Subject field
`pdftex`		use pdfTeX backend
`pdftitle`	*empty*	text for PDF Title field
`pdftoolbar`	*true*	make PDF toolbar visible
`pdfview`	*empty*	PDF 'view' when on link traversal
`pdfwindowui`	*true*	make PDF user interface elements visible
`plainpages`	*true*	do page number anchors as plain arabic
`ps2pdf`		use ps2pdf backend
`raiselinks`	*false*	raise up links (for HyperTeX backend)

`runbordercolor`	*0 .7 .7*	color of border around 'run' links
`setpagesize`	*true*	set page size by special driver commands
`tex4ht`		use TEX4ht backend
`textures`		use Textures backend
`unicode`		Unicode encoded pdf strings
`urlbordercolor`	*0 1 1*	color of border around URL links
`urlcolor`	*magenta*	color of URL links
`verbose`	*false*	be chatty
`vtex`		use VTeX backend

4 Additional user macros

If you need to make references to URLs, or write explicit links, the following low-level user macros are provided:

`\href{`*URL*`}{`*text*`}`

The *text* is made a hyperlink to the *URL*; this must be a full URL (relative to the base URL, if that is defined). The special characters # and ~ do *not* need to be escaped in any way.

`\url{`*URL*`}|`

Equivalent to `\href{`*URL*`}{`*URL*`}`.

`\nolinkurl{`*URL*`}|`

Write *URL* as plain text, without creating a hyperlink.

`\hyperbaseurl{`*URL*`}`

A base *URL* is established, which is prepended to other specified URLs, to make it easier to write portable documents.

`\hyperimage{`*imageURL*`}`

The image referenced by the *URL* is inserted.

`\hyperdef{`*category*`}{`*name*`}{`*text*`}`

A target area of the document (the *text*) is marked, and given the name *category.name*

`\hyperref{`*URL*`}{`*category*`}{`*name*`}{`*text*`}`

text is made into a link to *URL#category.name*

`\hyperlink{`*name*`}{`*text*`}`

`\hypertarget{`*name*`}{`*text*`}`

A simple internal link is created with `\hypertarget`, with two parameters of an anchor *name*, and

anchor *text*. `\hyperlink` has two arguments, the name of a hypertext object defined somewhere by `\hypertarget`, and the *text* which be used as the link on the page.

Note that in HTML parlance, the `\hyperlink` command inserts a notional # in front of each link, making it relative to the current testdocument; `\href` expects a full URL.

`\autoref{`*label*`}`

This is a replacement for the usual `\ref` command that places a contextual label in front of the reference. This gives your users a bigger target to click for hyperlinks (e.g. 'section 2' instead of merely the number '2').

The label is worked out from the context of the original `\label` command by hyperref by using the macros listed below (shown with their default values). The macros can be redefined in documents using `\renewcommand`; note that some of these macros are already defined in the standard document classes. The mixture of lowercase and uppercase initial letters is deliberate and corresponds to the author's practice.

For each macro below, hyperref checks `\*autorefname` before `\*name`. For instance, it looks for `\figureautorefname` before `\figurename`.

Macro	*Default*
`\figurename`	Figure
`\tablename`	Table
`\partname`	Part
`\appendixname`	Appendix
`\equationname`	Equation
`\Itemname`	item
`\Chaptername`	chapter
`\sectionname`	section
`\subsectionname`	subsection
`\subsubsectionname`	subsubsection
`\paragraphname`	paragraph
`\Hfootnotename`	footnote
`\AMSname`	Equation
`\theoremname`	Theorem

For instances where you want a reference to use the correct counter, but not to create a link, there are two starred forms:

`\ref*{`*label*`}`

`\pageref*{`*label*`}`

A typical use would be to write

```
\hyperref{other}{that nice section (\ref*{other}) we read before}
```

We want `\ref*{other}` to generate the correct number, but not to form a link, since we do this ourselves with `\hyperref`.

`\pdfstringdef{`*macroname*`}{`*TEXstring*`}`

`\pdfstringdef` returns a macro containing the PDF string. (Currently this is done globally,

but do not rely on it.) All the following tasks, definitions and redefinitions are made in a group to keep them local:

- Switching to PD1 or PU encoding
- Defining the Şoctal sequence commandsŤ (`\345`): `\edef\3{\string\3}`
- Special glyphs of TEX: `\{`, `\%`, `\&`, `\space`, `\dots`, etc.
- National glyphs (german.sty, french.sty, etc.)
- Logos: `\TeX`, `\eTeX`, `\MF`, etc.
- Disabling commands that do not provide useful functionality in bookmarks: `\label`, `\index`, `\glossary`, `\discretionary`, `\def`, `\let`, etc.
- LATEXŠs font commands like `\textbf`, etc.
- Support for `\xspace` provided by the xspace package

In addition, parentheses are protected to avoid the danger of unsafe unbalanced parentheses in the PDF string. For further details, see Heiko Oberdiek's EuroTEX paper distributed with hyperref.

4.1 Replacement macros

hyperref takes the text for bookmarks from the arguments of commands like `\section`, which can contain things like math, colors, or font changes, none of which will display in bookmarks as is.

`\texorpdfstring{`*TEXstring*`}{`*PDFstring*`}`

For example,

```
\section{Pythagoras:
\texorpdfstring{$ a^2 + b^2 = c^2 $}{%
a\texttwosuperior\ + b\texttwosuperior\ =
c\texttwosuperior}}
\section{\texorpdfstring{\textcolor{red}}{}{Red} Mars}
```

`\pdfstringdef` executes the hook before it expands the string. Therefore, you can use this hook to perform additional tasks or to disable additional commands.

```
\expandafter\def\expandafter\pdfstringdefPreHook
\expandafter{%
\pdfstringdefPreHook
\renewcommand{\mycommand}[1]{}%
}
```

However, for disabling commands, an easier way is via `\pdfstringdefDisableCommands`, which adds its argument to the definition of `\pdfstringdefPreHook` ('@' can here be used as letter in command names):

```
\pdfstringdefDisableCommands{%
\let~\textasciitilde
\def\url{\pdfstringdefwarn\url}%
\let\textcolor\@gobble
}
```

5 Acrobat-specific behavior

If you want to access the menu options of Acrobat Reader or Exchange, the following macro is provided in the appropriate drivers:

`\Acrobatmenu{`*menuoption*`}{`*text*`}`

The *text* is used to create a button which activates the appropriate *menuoption*. The following table lists the option names you can use—comparison of this with the menus in Acrobat Reader or Exchange will show what they do. Obviously some are only appropriate to Exchange.

File	Open, Close, Scan, Save, SaveAs, Optimizer:SaveAsOpt, Print, PageSetup, Quit
File→Import	ImportImage, ImportNotes, AcroForm:ImportFDF
File→Export	ExportNotes, AcroForm:ExportFDF
File→DocumentInfo	GeneralInfo, OpenInfo, FontsInfo, SecurityInfo, Weblink:Base, AutoIndex:DocInfo
File→Preferences	GeneralPrefs, NotePrefs, FullScreenPrefs, Weblink:Prefs, AcroSearch:Preferences(Windows) or, AcroSearch:Prefs(Mac), Cpt:Capture
Edit	Undo, Cut, Copy, Paste, Clear, SelectAll, Ole:CopyFile, TouchUp:TextAttributes, TouchUp:FitTextToSelection, TouchUp:ShowLineMarkers, TouchUp:ShowCaptureSuspects, TouchUp:FindSuspect, Properties
Edit→Fields	AcroForm:Duplicate, AcroForm:TabOrder
Document	Cpt:CapturePages, AcroForm:Actions, CropPages, RotatePages, InsertPages, ExtractPages, ReplacePages, DeletePages, NewBookmark, SetBookmarkDest, CreateAllThumbs, DeleteAllThumbs
View	ActualSize, FitVisible, FitWidth, FitPage, ZoomTo, FullScreen, FirstPage, PrevPage, NextPage, LastPage, GoToPage, GoBack, GoForward, SinglePage, OneColumn, TwoColumns, ArticleThreads, PageOnly, ShowBookmarks, ShowThumbs
Tools	Hand, ZoomIn, ZoomOut, SelectText, SelectGraphics, Note, Link, Thread, AcroForm:Tool, Acro_Movie:MoviePlayer, TouchUp:TextTool, Find, FindAgain, FindNextNote, CreateNotesFile
Tools→Search	AcroSrch:Query, AcroSrch:Indexes, AcroSrch:Results, AcroSrch:Assist, AcroSrch:PrevDoc, AcroSrch:PrevHit, AcroSrch:NextHit, AcroSrch:NextDoc
Window	ShowHideToolBar, ShowHideMenuBar, ShowHideClipboard, Cascade, TileHorizontal, TileVertical, CloseAll
Help	HelpUserGuide, HelpTutorial, HelpExchange, HelpScan, HelpCapture, HelpPDFWriter, HelpDistiller, HelpSearch, HelpCatalog, HelpReader, Weblink:Home
Help(Windows)	About

6 PDF and HTML forms

You must put your fields inside a `Form` environment (only one per file).

There are six macros to prepare fields:

`\TextField[`*parameters*`]{`*label*`}`

`\CheckBox[`*parameters*`]{`*label*`}`

`\ChoiceMenu[`*parameters*`]{`*label*`}{`*choices*`}`

`\PushButton[`*parameters*`]{`*label*`}`

`\Submit[`*parameters*`]{`*label*`}`

`\Reset[`*parameters*`]{`*label*`}`

The way forms and their labels are laid out is determined by:

`\LayoutTextField{`*label*`}{`*field*`}`

`\LayoutChoiceField{`*label*`}{`*field*`}`

`\LayoutCheckboxField{`*label*`}{`*field*`}`

These macros default to #1 #2

What is actually shown in as the field is determined by:

`\MakeRadioField{`*width*`}{`*height*`}`

`\MakeCheckField{`*width*`}{`*height*`}`

`\MakeTextField{`*width*`}{`*height*`}`

`\MakeChoiceField{`*width*`}{`*height*`}`

`\MakeButtonField{`*text*`}`

These macros default to `\vbox to #2{\hbox to #1{\hfill}\vfill}`, except the last, which defaults to #1; it is used for buttons, and the special `\Submit` and `\Reset` macros.

You may also want to redefine the following macros:

```
\def\DefaultHeightofSubmit{12pt}
\def\DefaultWidthofSubmit{2cm}
\def\DefaultHeightofReset{12pt}
\def\DefaultWidthofReset{2cm}
\def\DefaultHeightofCheckBox{0.8\baselineskip}
\def\DefaultWidthofCheckBox{0.8\baselineskip}
\def\DefaultHeightofChoiceMenu{0.8\baselineskip}
\def\DefaultWidthofChoiceMenu{0.8\baselineskip}
\def\DefaultHeightofText{\baselineskip}
\def\DefaultWidthofText{3cm}
```

6.1 Forms environment parameters

`action`	*URL*	The URL that will receive the form data if a Submit button is included in the form
`encoding`	*name*	The encoding for the string set to the URL; FDF-encoding is usual, and `html` is the only valid value
`method`	*name*	Used only when generating HTML; values can be `post` or `get`

6.2 Forms optional parameters

Note that all colors must be expressed as RGB triples, in the range 0..1 (i.e. `color=0 0 0.5`)

`accesskey`	key		(as per HTML)
`align`	number	*0*	alignment within text field; 0 is left-aligned, 1 is centered, 2 is right-aligned.
`backgroundcolor`			color of box
`bordercolor`			color of border
`bordersep`			box border gap
`borderwidth`			width of box border
`calculate`			JavaScript code to calculate the value of the field
`charsize`	dimen		font size of field text
`checked`	boolean	*false*	whether option selected by default
`color`			color of text in box
`combo`	boolean	*false*	choice list is 'combo' style
`default`			default value
`disabled`	boolean	*false*	field disabled
`format`			JavaScript code to format the field
`height`	dimen		height of field box
`hidden`	boolean	*false*	field hidden
`ketstroke`			JavaScript code to control the keystrokes on entry
`maxlen`	number	*0*	number of characters allowed in text field
`menulength`	number	*4*	number of elements shown in list
`multiline`	boolean	*false*	whether text box is multiline
`name`	name		name of field (defaults to label)
`onblur`			JavaScript code
`onchange`			JavaScript code
`onclick`			JavaScript code

ondblclick			JavaScript code
onfocus			JavaScript code
onkeydown			JavaScript code
onkeypress			JavaScript code
onkeyup			JavaScript code
onmousedown			JavaScript code
onmousemove			JavaScript code
onmouseout			JavaScript code
onmouseover			JavaScript code
onmouseup			JavaScript code
onselect			JavaScript code
password	boolean	*false*	text field is 'password' style
popdown	boolean	*false*	choice list is 'popdown' style
radio	boolean	*false*	choice list is 'radio' style
readonly	boolean	*false*	field is readonly
tabkey			(as per HTML)
validate			JavaScript code to validate the entry
value			initial value
width	dimen		width of field box

7 Defining a new driver

A hyperref driver has to provide definitions for eight macros:

1. `\hyper@anchor`
2. `\hyper@link`
3. `\hyper@linkfile`
4. `\hyper@linkurl`
5. `\hyper@anchorstart`
6. `\hyper@anchorend`
7. `\hyper@linkstart`
8. `\hyper@linkend`

The draft option defines the macros as follows

```
\let\hyper@@anchor\@gobble
\gdef\hyper@link##1##2##3{##3}%
\def\hyper@linkurl##1##2{##1}%
\def\hyper@linkfile##1##2##3{##1}%
\let\hyper@anchorstart\@gobble
\let\hyper@anchorend\@empty
\let\hyper@linkstart\@gobbletwo
\let\hyper@linkend\@empty
```

8 Special support for other packages

hyperref aims to cooperate with other packages, but there are several possible sources for conflict, such as

- Packages that manipulate the bibliographic mechanism. Peter William's harvard package is supported. However, the recommended package is Patrick Daly's natbib package that has specific hyperref hooks to allow reliable interaction. This package covers a very wide variety of layouts and citation styles, all of which work with hyperref.

- Packages that typeset the contents of the `\label` and `\ref` macros, such as showkeys. Since the hyperref package redefines these commands, you must set `implicit=false` for these packages to work.
- Packages that do anything serious with the index.

The hyperref package is distributed with variants on two useful packages designed to work especially well with it. These are xr and minitoc, which support crossdocument links using LaTeX's normal `\label/\ref` mechanisms and per-chapter tables of contents, respectively.

9 History and acknowledgments

The original authors of hyperbasics.tex and hypertex.sty, from which this package descends, are Tanmoy Bhattacharya (`tanmoy@qcd.lanl.gov`) and Thorsten Ohl (`thorsten.ohl@physik.th-darmstadt.de`). hyperref started as a simple port of their work to LaTeX 2_ε standards, but eventually I rewrote nearly everything, because I didn't understand a lot of the original, and was only interested in getting it to work with LaTeX. I would like to thank Arthur Smith, Tanmoy Bhattacharya, Mark Doyle, Paul Ginsparg, David Carlisle, T. V. Raman and Leslie Lamport for comments, requests, thoughts and code to get the package into its first useable state. Various other people are mentioned at the point in the source where I had to change the code in later versions because of problems they found.

Tanmoy found a great many of the bugs, and (even better) often provided fixes, which has made the package more robust. The days spent on RevTeX are entirely due to him! The investigations of Bill Moss (`bmoss@math.clemson.edu`) into the later versions including native PDF support uncovered a good many bugs, and his testing is appreciated. Hans Hagen (`pragma@pi.net`) provided a lot of insight into PDF.

Berthold Horn provided help, encouragement and sponsorship for the dvipsone and dviwindo drivers. Sergey Lesenko provided the changes needed for dvipdf, and Hàn Thế Thành supplied all the information needed for pdftex. Patrick Daly kindly updated his natbib package to allow easy integration with hyperref. Michael Mehlich's hyper package (developed in parallel with hyperref) showed me solutions for some problems. Hopefully the two packages will combine one day.

The forms creation section owes a great deal to: T. V. Raman, for encouragement, support and ideas; Thomas Merz, whose book *Web Publishing with Acrobat/PDF* provided crucial insights; D. P. Story, whose detailed article about pdfmarks and forms solved many practical problems; and Hans Hagen, who explained how to do it in pdftex.

Steve Dandy recreated the manual source in July 2003 after it had been lost.

Especial extra thanks to David Carlisle for the backref module, the ps2pdf and dviwindo support, frequent general rewrites of my bad code, and for working on changes to the xr package to suit hyperref.