

# Hypertext marks in L<sup>A</sup>T<sub>E</sub>X: a manual for hyperref

Sebastian Rahtz      Heiko Oberdiek

September 2019

## Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Implicit behavior</b>	<b>5</b>
<b>3</b>	<b>Package options</b>	<b>5</b>
3.1	General options . . . . .	6
3.2	Options for destination names . . . . .	6
3.3	Configuration options . . . . .	8
3.4	Backend drivers . . . . .	9
3.5	Extension options . . . . .	9
3.6	PDF-specific display options . . . . .	11
3.7	PDF display and information options . . . . .	12
3.8	Option <code>pdfinfo</code> . . . . .	14
3.9	Big alphabetical list . . . . .	14
<b>4</b>	<b>Additional user macros</b>	<b>17</b>
4.1	Bookmark macros . . . . .	21
4.1.1	Setting bookmarks . . . . .	21
4.1.2	Replacement macros . . . . .	21
4.2	Utility macros . . . . .	22
<b>5</b>	<b>New Features</b>	<b>22</b>
5.1	Option <code>'pdfinkmargin'</code> . . . . .	22
5.2	Field option <code>'calculatesortkey'</code> . . . . .	23
5.3	Option <code>'localanchorname'</code> . . . . .	23
5.4	Option <code>'customdriver'</code> . . . . .	23
5.5	Option <code>'psdextra'</code> . . . . .	23
5.6	<code>\XeTeXLinkBox</code> . . . . .	24
5.7	<code>\IfHyperBooleanExists</code> and <code>\IfHyperBoolean</code> . . . . .	24
5.8	<code>\unichar</code> . . . . .	24
5.9	<code>\ifpdfstringunicode</code> . . . . .	25
5.10	Customizing index style file with <code>\nohyperpage</code> . . . . .	26
5.11	Experimental option <code>'ocgcolorlinks'</code> . . . . .	26
5.12	Option <code>'pdfa'</code> . . . . .	26
5.13	Option <code>'linktoc'</code> added . . . . .	27
5.14	Option <code>'pdfnewwindow'</code> changed . . . . .	27
5.15	Flag options for PDF forms . . . . .	27
5.16	Option <code>'pdfversion'</code> . . . . .	29
5.17	Field option <code>'name'</code> . . . . .	30

5.18	Option ‘pdfencoding’	30
5.19	Color options/package hicolor	30
5.20	Option pdfusetitle	30
5.21	Starred form of \autoref	30
5.22	Link border style	30
5.23	Option ”bookmarksdepth”	31
5.24	Option ”pdfescapeform”	31
5.25	Default driver setting	31
5.26	Backref entries	32
5.27	\phantomsection	33
<b>6</b>	<b>Acrobat-specific behavior</b>	<b>34</b>
<b>7</b>	<b>PDF and HTML forms</b>	<b>35</b>
7.1	Forms environment parameters	36
7.2	Forms optional parameters	36
<b>8</b>	<b>Defining a new driver</b>	<b>38</b>
<b>9</b>	<b>Special support for other packages</b>	<b>38</b>
9.1	Package Compatibility	38
9.1.1	algorithm	38
9.1.2	amsmath	39
9.1.3	amsrefs	39
9.1.4	arydshln, longtable	39
9.1.5	babel/magyar.ldf	39
9.1.6	babel/spanish.ldf	39
9.1.7	bibentry	39
9.1.8	bigfoot	40
9.1.9	chappg	40
9.1.10	cite	40
9.1.11	count1to	40
9.1.12	dblaccnt	40
9.1.13	easyeqn	40
9.1.14	ellipsis	41
9.1.15	float	41
9.1.16	endnotes	41
9.1.17	foiltex	41
9.1.18	footnote	41
9.1.19	geometry	41
9.1.20	IEEEtran.cls	42
9.1.21	index	42
9.1.22	lastpage	42
9.1.23	linguex	42
9.1.24	ltabptch	42
9.1.25	mathenv	42
9.1.26	minitoc-hyper	42
9.1.27	multind	42
9.1.28	natbib	42
9.1.29	nomenc1	43
9.1.30	parskip	43
9.1.31	prettyref	43

9.1.32	<code>ntheorem</code>	43
9.1.33	<code>setspace</code>	43
9.1.34	<code>sidecap</code>	44
9.1.35	<code>subfigure</code>	44
9.1.36	<code>titleref</code>	44
9.1.37	<code>tabularx</code>	44
9.1.38	<code>titlesec</code>	44
9.1.39	<code>ucs/utf8x.def</code>	44
9.1.40	<code>varioref</code>	45
9.1.41	<code>verse</code>	45
9.1.42	<code>vietnam</code>	46
9.1.43	<code>XeTeX</code>	46
<b>10</b>	<b>Limitations</b>	<b>47</b>
10.1	Wrapped/broken link support	47
10.2	Links across pages	47
10.3	Footnotes	47
<b>11</b>	<b>Hints</b>	<b>47</b>
11.1	Spaces in option values	47
11.2	Index with <code>makeindex</code>	48
11.3	Warning "bookmark level for unknown <foobar> defaults to 0"	49
11.4	Link anchors in figures	49
11.5	Additional unicode characters in bookmarks and pdf information entries:	49
11.6	Footnotes	50
11.7	Subordinate counters	51
<b>12</b>	<b>History and acknowledgments</b>	<b>51</b>
<b>13</b>	<b>GNU Free Documentation License</b>	<b>52</b>

## 1 Introduction

The package derives from, and builds on, the work of the HyperTeX project, described at <http://xxx.lanl.gov/hypertext/>. It extends the functionality of all the L<sup>A</sup>T<sub>E</sub>X 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 L<sup>A</sup>T<sub>E</sub>X Web Companion*, where you will find additional examples.

The HyperTeX specification<sup>1</sup> 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">`

<sup>1</sup>This is borrowed from an article by Arthur Smith.

**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 used to communicate to the DVI viewer the full (URL) location of the current document so that files specified by relative URLs 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 Hyper $\TeX$  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  $\TeX$ 4ht

**pdftex** pdf $\TeX$ , Hàn Thế Thành’s  $\TeX$  variant that writes PDF directly

**luatex** lua $\TeX$ , Unicode  $\TeX$  variant that writes PDF directly

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

**dvipdfmx** produces `\special` commands for driver *dvipdfmx*, a successor of *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

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

**xetex** produces `\special` commands for Xe $\TeX$

Output from *dvips* or *dvipsone* must be processed using Acrobat Distiller to obtain a PDF file.<sup>2</sup> The result is generally preferable to that produced by using the *hypertex* driver, and then

<sup>2</sup>Make sure you turn off the partial font downloading supported by *dvips* and *dvipsone* in favor of Distiller’s own system.

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`.

**driverfallback** If a driver is not given and cannot be autodetected, then use the driver option, given as value to this option `driverfallback`. Example:

```
driverfallback=dvipdfm
```

Autodetected drivers (`pdftex`, `xetex`, `vtex`, `vtexpdfmark`) are recognized from within TeX and therefore cannot be given as value to option `driverfallback`. However a DVI driver program is run after the TeX run is finished. Thus it cannot be detected at TeX macro level. Then package `hyperref` uses the driver, given by `driverfallback`. If the driver is already specified or can be autodetected, then option `driverfallback` is ignored.

## 2 Implicit behavior

This package can be used with more or less any normal L<sup>A</sup>T<sub>E</sub>X 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 L<sup>A</sup>T<sub>E</sub>X 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.

Note however that some options (for example `unicode`) can only be used as package options, and not in `\hypersetup` as the option settings are processed as the package is read.

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`:

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

As seen in the previous example, information entries (`pdftitle`, `pdfauthor`, ...) should be set after the package is loaded. Otherwise  $\LaTeX$  expands the values of these options prematurely. Also  $\LaTeX$  strips spaces in options. Especially option ‘`pdfborder`’ requires some care. Curly braces protect the value, if given as package option. They are not necessary in `\hypersetup`.

```
\usepackage[pdfborder={0 0 0}]{hyperref}
\hypersetup{pdfborder=0 0 0}
```

Package ‘`kvoptions-patch`’ patches  $\LaTeX$  to make it aware of key value options and to prevent premature value expansions.

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.

<code>draft</code>	boolean	<i>false</i>	all hypertext options are turned off
<code>final</code>	boolean	<i>true</i>	all hypertext options are turned on
<code>debug</code>	boolean	<i>false</i>	extra diagnostic messages are printed in the log file
<code>verbose</code>	boolean	<i>false</i>	same as <code>debug</code>
<code>implicit</code>	boolean	<i>true</i>	redefines $\LaTeX$ internals
<code>setpagesize</code>	boolean	<i>true</i>	sets page size by special driver commands

### 3.2 Options for destination names

Destinations names (also anchor, target or link names) are internal names that identify a position on a page in the document. They are used in link targets for inner document links or the bookmarks, for example.

Usually anchor are set, if `\refstepcounter` is called. Thus there is a counter name and value. Both are used to construct the destination name. By default the counter value follows the counter name separated by a dot. Example for the fourth chapter:

```
chapter.4
```

This scheme is used by:

`\autoref` displays the description label for the reference depending on the counter name.

`\hyperpage` is used by the index to get page links. Page anchor setting (`pageanchor`) must not be turned off.

It is very important that the destination names are unique, because two destinations must not share the same name. The counter value `\the<counter>` is not always unique for the counter. For example, table and figures can be numbered inside the chapter without having the chapter number in their number. Therefore `hyperref` has introduced `\theH<counter>` that allows a unique counter value without messing up with the appearance of the counter number. For example, the number of the second table in the third chapter might be printed as 2, the result of `\thetable`. But the destination name `table.2.4` is unique because it has used `\theHtable` that gives 2.4 in this case.

Often the user do not need to set `\theH<counter>`. Defaults for standard cases (chapter, ...) are provided. And after `hyperref` is loaded, new counters with parent counters also define `\theH<counter>` automatically, if `\newcounter`, `\addtoreset` or `\numberwithin` of package `amsmath` are used.

Usually problems with duplicate destination names can be solved by an appropriate definition of `\theH<counter>`. If option `hypertexnames` is disabled, then a unique artificial number is used instead of the counter value. In case of page anchors the absolute page anchor is used. With option `plainpages` the page anchors use the arabic form. In both latter cases `\hyperpage` for index links is affected and might not work properly.

If an unnumbered entity gets an anchor (starred forms of chapters, sections, ...) or `\phantomsection` is used, then the dummy counter name `section*` and an artificial unique number is used.

If the final PDF file is going to be merged with another file, than the destination names might clash, because both documents might contain `chapter.1` or `page.1`. Also `hyperref` sets anchor with name `Doc-Start` at the begin of the document. This can be resolved by redefining `\HyperDestNameFilter`. Package `hyperref` calls this macro each time, it uses a destination name. The macro must be expandable and expects the destination name as only argument. As example, the macro is redefined to add a prefix to all destination names:

```
\renewcommand*\HyperDestNameFilter}[1]{\jobname-#1}
```

In document `docA` the destination name `chapter.2` becomes `docA-chapter.2`.

Destination names can also be used from the outside in URIs(, if the driver has not removed or changed them), for example:

```
http://somewhere/path/file.pdf#nameddest=chapter.4
```

However using a number seems unhappy. If another chapter is added before, the number changes. But it is very difficult to pass a new name for the destination to the anchor setting process that is usually deep hidden in the internals. The first name of `\label` after the anchor setting seems a good approximation:

```
\section{Introduction}
\label{intro}
```

Option `destlabel` checks for each `\label`, if there is a new destination name active and replaces the destination name by the label name. Because the destination name is already in use because of the anchor setting, the new name is recorded in the `.aux` file and used in the subsequent `LATEX` run. The renaming is done by a redefinition of `\HyperDestNameFilter`. That leaves the old destination names intact (e.g., they are needed for `\autoref`). This redefinition is also available as `\HyperDestLabelReplace`, thus that an own redefinition can use it. The following example also adds a prefix for *all* destination names:

```
\renewcommand*\HyperDestNameFilter}[1]{%
  \jobname-\HyperDestLabelReplace{#1}%
}
```

The other case that only files prefixed that do not have a corresponding `\label` is more complicate, because `\HyperDestLabelReplace` needs the unmodified destination name as argument. This is solved by an expandable string test (`\pdfstrcmp` of pdfTeX or `\strcmp` of XeTeX, package `pdftexcmds` also supports LuaTeX):

```
\usepackage{pdftexcmds}
\makeatletter
\renewcommand*{\HyperDestNameFilter}[1]{%
  \ifcase\pdf@strcmp{#1}{\HyperDestLabelReplace{#1}} %
    \jobname-#1%
  \else
    \HyperDestLabelReplace{#1}%
  \fi
}
\makeatother
```

With option `destlabel` destinations can also named manually, if the destination is not yet renamed:

```
\HyperDestRename{<destination>}{<newname>}
```

Hint: Anchors can also be named and set by `\hypertarget`.

<code>destlabel</code>	boolean	<i>false</i>	destinations are named by first <code>\label</code> after anchor creation
<code>hypertexnames</code>	boolean	<i>true</i>	use guessable names for links
<code>naturalnames</code>	boolean	<i>false</i>	use L <sup>A</sup> T <sub>E</sub> X-computed names for links
<code>plainpages</code>	boolean	<i>false</i>	Forces page anchors to be named by the Arabic form of the page number, rather than the formatted form.

### 3.3 Configuration options

<code>raiselinks</code>	boolean	<i>true</i>	In the hypertext driver, the height of links is normally calculated by the driver as simply the base line of contained text; this options forces <code>\special</code> commands to reflect the real height of the link (which could contain a graphic)
<code>breaklinks</code>	boolean	<i>false</i>	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.
<code>pageanchor</code>	boolean	<i>true</i>	Determines whether every page is given an implicit anchor at the top left corner. If this is turned off, <code>\printindex</code> will not contain valid hyperlinks.
<code>nesting</code>	boolean	<i>false</i>	Allows links to be nested; no drivers currently support this.

Note for option `breaklinks`: The correct value is automatically set according to the driver features. It can be overwritten for drivers that do not support broken links. However, at any case,



the link area will be wrong and displaced.

### 3.4 Backend drivers

If no driver is specified, the package tries to find a driver in the following order:

1. Autodetection, some  $\TeX$  processors can be detected at  $\TeX$  macro level (pdf $\TeX$ , Xe $\TeX$ , V $\TeX$ ).
2. Option `driverfallback`. If this option is set, its value is taken as driver option.
3. Macro `\Hy@defaultdriver`. The macro takes a driver file name (without file extension).
4. Package default is `hypertex`.

Many distributions are using a driver file `hypertex.cfg` that define `\Hy@defaultdriver` with `hdvips`. This is recommended because driver `dvips` provides much more features than `hypertex` for PDF generation.

<code>driverfallback</code>	Its value is used as driver option if the driver is not given or autodetected.
<code>dvipdfm</code>	Sets up <code>hyperref</code> for use with the <code>dvipdfm</code> driver.
<code>dvipdfmx</code>	Sets up <code>hyperref</code> for use with the <code>dvipdfmx</code> driver.
<code>dvips</code>	Sets up <code>hyperref</code> for use with the <code>dvips</code> driver.
<code>dvipsone</code>	Sets up <code>hyperref</code> for use with the <code>dvipsone</code> driver.
<code>dviwindo</code>	Sets up <code>hyperref</code> for use with the <code>dviwindo</code> Windows previewer.
<code>hypertex</code>	Sets up <code>hyperref</code> for use with the Hyper $\TeX$ -compliant drivers.
<code>latex2html</code>	Redefines a few macros for compatibility with <code>latex2html</code> .
<code>nativepdf</code>	An alias for <code>dvips</code>
<code>pdfmark</code>	An alias for <code>dvips</code>
<code>pdftex</code>	Sets up <code>hyperref</code> for use with the <code>pdftex</code> program.
<code>ps2pdf</code>	Redefines a few macros for compatibility with Ghostscript's PDF writer, otherwise identical to <code>dvips</code> .
<code>tex4ht</code>	For use with $\TeX$ 4ht
<code>textures</code>	For use with Textures
<code>vtex</code>	For use with MicroPress' V $\TeX$ ; the PDF and HTML backends are detected automatically.
<code>vtexpdfmark</code>	For use with V $\TeX$ 's PostScript backend.
<code>xetex</code>	For use with Xe $\TeX$ (using backend for <code>dvipdfm</code> ).

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}
```

### 3.5 Extension options

<code>extension</code>	<code>text</code>	Set the file extension (e.g. <code>dvi</code> ) which will be appended to file links created if you use the <code>xr</code> package.
------------------------	-------------------	--

hyperfigures	boolean		
backref	text	<i>false</i>	Adds ‘backlink’ text to the end of each item in the bibliography, as a list of section numbers. This can only work properly <i>if</i> there is a blank line after each <code>\bibitem</code> . Supported values are <b>section</b> , <b>slide</b> , <b>page</b> , <b>none</b> , or <b>false</b> . If no value is given, <b>section</b> is taken as default.
pagebackref	boolean	<i>false</i>	Adds ‘backlink’ text to the end of each item in the bibliography, as a list of page numbers.
hyperindex	boolean	<i>true</i>	Makes the page numbers of index entries into hyperlinks. Relays on unique page anchors ( <code>pageanchor</code> , ...)
pageanchors and plainpages=false. hyperfootnotes	boolean	<i>true</i>	Makes the footnote marks into hyperlinks to the footnote text. Easily broken ...
encap			Sets encap character for hyperindex
linktoc	text	<i>section</i>	make text ( <b>section</b> ), page number ( <b>page</b> ), both ( <b>all</b> ) or nothing ( <b>none</b> ) be link on TOC, LOF and LOT
linktocpage	boolean	<i>false</i>	make page number, not text, be link on TOC, LOF and LOT
breaklinks	boolean	<i>false</i>	allow links to break over lines by making links over multiple lines into PDF links to the same target
colorlinks	boolean	<i>false</i>	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. Unlike colored boxes, the colored text remains when printing.
linkcolor	color	<i>red</i>	Color for normal internal links.
anchorcolor	color	<i>black</i>	Color for anchor text.
citecolor	color	<i>green</i>	Color for bibliographical citations in text.
filecolor	color	<i>cyan</i>	Color for URLs which open local files.
menucolor	color	<i>red</i>	Color for Acrobat menu items.
runcolor	color	<i>filecolor</i>	Color for run links (launch annotations).
urlcolor	color	<i>magenta</i>	Color for linked URLs.
allcolors	color		Set all color options (without border and field options).
frenchlinks	boolean	<i>false</i>	Use small caps instead of color for links.

<code>hidelinks</code>			Hide links (removing color and border).
------------------------	--	--	---

Note that all color names must be defined before use, following the normal system of the standard L<sup>A</sup>T<sub>E</sub>X color package.

### 3.6 PDF-specific display options

<code>bookmarks</code>	boolean	<i>true</i>	A set of Acrobat bookmarks are written, in a manner similar to the table of contents, requiring two passes of L <sup>A</sup> T <sub>E</sub> X. Some postprocessing of the bookmark file (file extension <code>.out</code> ) may be needed to translate L <sup>A</sup> T <sub>E</sub> X codes, since bookmarks must be written in PDFEncoding. To aid this process, the <code>.out</code> file is not rewritten by L <sup>A</sup> T <sub>E</sub> X if it is edited to contain a line <code>\let\WriteBookmarks\relax</code>
<code>bookmarksopen</code>	boolean	<i>false</i>	If Acrobat bookmarks are requested, show them with all the subtrees expanded.
<code>bookmarksopenlevel</code>	parameter		level ( <code>\maxdimen</code> ) to which bookmarks are open
<code>bookmarksnumbered</code>	boolean	<i>false</i>	If Acrobat bookmarks are requested, include section numbers.
<code>bookmarkstype</code>	text	<i>toc</i>	to specify which ‘toc’ file to mimic
<code>CJKbookmarks</code>	boolean	<i>false</i>	This option should be used to produce CJK bookmarks. Package <code>hyperref</code> supports both normal and preprocessed mode of the CJK package; during the creation of bookmarks, it simply replaces CJK’s macros with special versions which expand to the corresponding character codes. Note that without the ‘unicode’ option of <code>hyperref</code> you get PDF files which actually violate the PDF specification because non-Unicode character codes are used – some PDF readers localized for CJK languages (most notably Acroread itself) support this. Also note that option ‘CJKbookmarks’ cannot be used together with option ‘unicode’. No mechanism is provided to translate non-Unicode bookmarks to Unicode; for portable PDF documents only Unicode encoding should be used.
<code>pdfhighlight</code>	name	<i>/I</i>	How link buttons behave when selected; <i>/I</i> is for inverse (the default); the other possibilities are <i>/N</i> (no effect), <i>/O</i> (outline), and <i>/P</i> (inset highlighting).
<code>citebordercolor</code>	RGB color	<i>0 1 0</i>	The color of the box around citations
<code>filebordercolor</code>	RGB color	<i>0 .5 .5</i>	The color of the box around links to files
<code>linkbordercolor</code>	RGB color	<i>1 0 0</i>	The color of the box around normal links
<code>menubordercolor</code>	RGB color	<i>1 0 0</i>	The color of the box around Acrobat menu links
<code>urlbordercolor</code>	RGB color	<i>0 1 1</i>	The color of the box around links to URLs
<code>runbordercolor</code>	RGB color	<i>0 .7 .7</i>	Color of border around ‘run’ links
<code>allbordercolors</code>			Set all border color options

<code>pdfborder</code>	<code>0 0 1</code>	The style of box around links; defaults to a box with lines of 1pt thickness, but the <code>colorlinks</code> 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  $\TeX$ . Since version 6.76a this is no longer true. Especially with the help of package `xcolor` the usual color specifications of package `(x)color` can be used. For further information see description of package `hicolor`.

The bookmark commands are stored in a file called `jobname.out`. The file 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
```

### 3.7 PDF display and information options

<code>baseurl</code>	URL		Sets the base URL of the PDF document
<code>pdfpagemode</code>	text	<i>empty</i>	Determines how the file is opening in Acrobat; the possibilities are <code>UseNone</code> , <code>UseThumbs</code> (show thumbnails), <code>UseOutlines</code> (show bookmarks), <code>FullScreen</code> , <code>UseOC</code> (PDF 1.5), and <code>UseAttachments</code> (PDF 1.6). If no mode is explicitly chosen, but the <code>bookmarks</code> option is set, <code>UseOutlines</code> is used.
<code>pdftitle</code>	text		Sets the document information Title field
<code>pdfauthor</code>	text		Sets the document information Author field
<code>pdfsubject</code>	text		Sets the document information Subject field
<code>pdfcreator</code>	text		Sets the document information Creator field
<code>addtopdfproducer</code>	text		Adds additional text to the document information Producer field
<code>pdfkeywords</code>	text		Sets the document information Keywords field
<code>pdftrapped</code>	text	<i>empty</i>	Sets the document information Trapped entry. Possible values are <code>True</code> , <code>False</code> and <code>Unknown</code> . An empty value means, the entry is not set.
<code>pdfinfo</code>	key value list	<i>empty</i>	Alternative interface for setting the document information.
<code>pdfview</code>	text	<i>XYZ</i>	Sets the default PDF ‘view’ for each link
<code>pdfstartpage</code>	text	<i>1</i>	Determines on which page the PDF file is opened.
<code>pdfstartview</code>	text	<i>Fit</i>	Set the startup page view
<code>pdfremotestartview</code>	text	<i>Fit</i>	Set the startup page view of remote PDF files
<code>pdfpagescrop</code>	n n n n		Sets the default PDF crop box for pages. This should be a set of four numbers
<code>pdfcenterwindow</code>	boolean	<i>false</i>	position the document window in the center of the screen
<code>pdfdirection</code>	text	<i>empty</i>	direction setting
<code>pdfdisplaydoctitle</code>	boolean	<i>false</i>	display document title instead of file name in title bar
<code>pdfduplex</code>	text	<i>empty</i>	paper handling option for print dialog
<code>pdffitwindow</code>	boolean	<i>false</i>	resize document window to fit document size
<code>pdflang</code>	text	<i>relax</i>	PDF language identifier (RFC 3066)

<code>pdfmenubar</code>	boolean	<i>true</i>	make PDF viewer's menu bar visible
<code>pdfnewwindow</code>	boolean	<i>false</i>	make links that open another PDF file start a new window
<code>pdfnonfullscreenpagemode</code>	boolean	<i>empty</i>	page mode setting on exiting full-screen mode
<code>pdfnumcopies</code>	integer	<i>empty</i>	number of printed copies
<code>pdfpagelayout</code>	text	<i>empty</i>	set layout of PDF pages
<code>pdfpagelabels</code>	boolean	<i>true</i>	set PDF page labels
<code>pdfpagetransition</code>	text	<i>empty</i>	set PDF page transition style
<code>pdfpicktraybypdfsize</code>	text	<i>empty</i>	set option for print dialog
<code>pdfprintarea</code>	text	<i>empty</i>	set /PrintArea of viewer preferences
<code>pdfprintclip</code>	text	<i>empty</i>	set /PrintClip of viewer preferences
<code>pdfprintpagerange</code>	n n (n n)*	<i>empty</i>	set /PrintPageRange of viewer preferences
<code>pdfprintscaling</code>	text	<i>empty</i>	page scaling option for print dialog (option /PrintScaling of viewer preferences, PDF 1.6); valid values are <code>None</code> and <code>AppDefault</code>
<code>pdftoolbar</code>	boolean	<i>true</i>	make PDF toolbar visible
<code>pdfviewarea</code>	text	<i>empty</i>	set /ViewArea of viewer preferences
<code>pdfviewclip</code>	text	<i>empty</i>	set /ViewClip of viewer preferences
<code>pdfwindowui</code>	boolean	<i>true</i>	make PDF user interface elements visible
<code>unicode</code>	boolean	<i>false</i>	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. The unit is bp and the origin is in the lower left corner. See also `\hypercalcbp` that is explained on page 22. pdf $\TeX$  works by supplying default values for XYZ (horizontal  $\times$  vertical  $\times$  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`, `pdfstartview` and `pdfremotestartview` parameters.

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

<code>TwoColumnRight</code>	Displays the document in two columns, odd-numbered pages to the right.
<code>TwoPageLeft</code>	Displays two pages, odd-numbered pages to the left (since PDF 1.5).
<code>TwoPageRight</code>	Displays two pages, odd-numbered pages to the right (since PDF 1.5).

Finally, the `pdfpagetransition` can be one of the following values, where  $/Di$  stands for direction of motion in degrees, generally in  $90^\circ$  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 ( $/O$ ).

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

### 3.8 Option `pdfinfo`

The information entries can be set using `pdftitle`, `pdfsubject`, .... Option `pdfinfo` provides an alternative interface. It takes a key value list. The key names are the names that appear in the PDF information dictionary directly. Known keys such as `Title`, `Subject`, `Trapped` and other are mapped to options `pdftitle`, `subject`, `trapped`, ...Unknown keys are added to the information dictionary. Their values are text strings (see PDF specification). Example:

```
\hypersetup{
  pdfinfo={
    Title={My Title},
    Subject={My Subject},
    NewKey={Foobar},
    % ...
  }
}
```

### 3.9 Big alphabetical list

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

<code>anchorcolor</code>	<i>black</i>	set color of anchors
<code>backref</code>	<i>false</i>	do bibliographical back references
<code>baseurl</code>	<i>empty</i>	set base URL for document
<code>bookmarks</code>	<i>true</i>	make bookmarks
<code>bookmarksnumbered</code>	<i>false</i>	put section numbers in bookmarks
<code>bookmarksopen</code>	<i>false</i>	open up bookmark tree
<code>bookmarksopenlevel</code>	<code>\maxdimen</code>	level to which bookmarks are open
<code>bookmarkstype</code>	<i>toc</i>	to specify which 'toc' file to mimic
<code>breaklinks</code>	<i>false</i>	allow links to break over lines

CJKbookmarks	<i>false</i>	to produce CJK bookmarks
citebordercolor	<i>0 1 0</i>	color of border around cites
citecolor	<i>green</i>	color of citation links
colorlinks	<i>false</i>	color links
	<i>true</i>	( <code>tex4ht</code> , <code>dviwindo</code> )
debug	<i>false</i>	provide details of anchors defined; same as <code>verbose</code>
destlabel	<i>false</i>	destinations are named by the first <code>\label</code> after the anchor creation
draft	<i>false</i>	do not do any hyperlinking
dvipdfm		use <code>dvipdfm</code> backend
dvipdfmx		use <code>dvipdfmx</code> backend
dvips		use <code>dvips</code> backend
dvipsone		use <code>dvipsone</code> backend
dviwindo		use <code>dviwindo</code> backend
encap		to set <code>encap</code> character for hyperindex
extension	<i>dvi</i>	suffix of linked files
filebordercolor	<i>0 .5 .5</i>	color of border around file links
filecolor	<i>cyan</i>	color of file links
final	<i>true</i>	opposite of option <code>draft</code>
frenchlinks	<i>false</i>	use small caps instead of color for links
hyperfigures	<i>false</i>	make figures hyper links
hyperfootnotes	<i>true</i>	set up hyperlinked footnotes
hyperindex	<i>true</i>	set up hyperlinked indices
hypertex		use <code>HyperTeX</code> backend
hypertexnames	<i>true</i>	use guessable names for links
implicit	<i>true</i>	redefine <code>L<sup>A</sup>T<sub>E</sub>X</code> internals
latex2html		use <code>L<sup>A</sup>T<sub>E</sub>X2HTML</code> backend
linkbordercolor	<i>1 0 0</i>	color of border around links
linkcolor	<i>red</i>	color of links
linktoc	<i>section</i>	make text be link on TOC, LOF and LOT
linktocpage	<i>false</i>	make page number, not text, be link on TOC, LOF and LOT
menubordercolor	<i>1 0 0</i>	color of border around menu links
menucolor	<i>red</i>	color for menu links
nativepdf	<i>false</i>	an alias for <code>dvips</code>
naturalnames	<i>false</i>	use <code>L<sup>A</sup>T<sub>E</sub>X</code> -computed names for links
nesting	<i>false</i>	allow nesting of links
pageanchor	<i>true</i>	put an anchor on every page
pagebackref	<i>false</i>	backreference by page number
pdfauthor	<i>empty</i>	text for PDF Author field
pdfborder	<i>0 0 1</i>	width of PDF link border
	<i>0 0 0</i>	( <code>colorlinks</code> )
pdfcenterwindow	<i>false</i>	position the document window in the center of the screen
pdfcreator	<i>LaTeX with hyperref</i>	text for PDF Creator field
pdfdirection	<i>empty</i>	direction setting
pdfdisplaydoctitle	<i>false</i>	display document title instead of file name in title bar
pdfduplex	<i>empty</i>	paper handling option for print dialog
pdffitwindow	<i>false</i>	resize document window to fit document size

pdfhighlight	<i>/I</i>	set highlighting of PDF links
pdfinfo	<i>empty</i>	alternative interface for setting document information
pdfkeywords	<i>empty</i>	text for PDF Keywords field
pdflang	<i>relax</i>	PDF language identifier (RFC 3066)
pdfmark	<i>false</i>	an alias for <code>dvips</code>
pdfmenubar	<i>true</i>	make PDF viewer's menu bar visible
pdfnewwindow	<i>false</i>	make links that open another PDF file start a new window
pdfnonfullscreenpagemode	<i>empty</i>	page mode setting on exiting full-screen mode
pdfnumcopies	<i>empty</i>	number of printed copies
pdfpagelayout	<i>empty</i>	set layout of PDF pages
pdfpagemode	<i>empty</i>	set default mode of PDF display
pdfpagelabels	<i>true</i>	set PDF page labels
pdfpagescrop	<i>empty</i>	set crop size of PDF document
pdfpagetransition	<i>empty</i>	set PDF page transition style
pdfpicktraybypdfsize	<i>empty</i>	set option for print dialog
pdfprintarea	<i>empty</i>	set <code>/PrintArea</code> of viewer preferences
pdfprintclip	<i>empty</i>	set <code>/PrintClip</code> of viewer preferences
pdfprintpagerange	<i>empty</i>	set <code>/PrintPageRange</code> of viewer preferences
pdfprintsampling	<i>empty</i>	page scaling option for print dialog
pdfproducer	<i>empty</i>	text for PDF Producer field
pdfremotestartview	<i>Fit</i>	starting view of remote PDF documents
pdfstartpage	<i>1</i>	page at which PDF document opens
pdfstartview	<i>Fit</i>	starting view of PDF document
pdfsubject	<i>empty</i>	text for PDF Subject field
pdfTeX		use pdfTeX backend
pdfTitle	<i>empty</i>	text for PDF Title field
pdftoolbar	<i>true</i>	make PDF toolbar visible
pdftrapped	<i>empty</i>	Sets the document information Trapped entry. Possible values are <b>True</b> , <b>False</b> and <b>Unknown</b> . An empty value means, the entry is not set.
pdfview	<i>XYZ</i>	PDF 'view' when on link traversal
pdfviewarea	<i>empty</i>	set <code>/ViewArea</code> of viewer preferences
pdfviewclip	<i>empty</i>	set <code>/ViewClip</code> of viewer preferences
pdfwindowui	<i>true</i>	make PDF user interface elements visible
plainpages	<i>false</i>	do page number anchors as plain Arabic
ps2pdf		use <code>ps2pdf</code> backend
raiselinks	<i>false</i>	raise up links (for HyperTeX backend)
runbordercolor	<i>0 .7 .7</i>	color of border around 'run' links
runcolor	<i>filecolor</i>	color of 'run' links
setpagesize	<i>true</i>	set page size by special driver commands
tex4ht		use TeX4ht backend
textures		use Textures backend
unicode	<i>false</i>	Unicode encoded pdf strings
urlbordercolor	<i>0 1 1</i>	color of border around URL links
urlcolor	<i>magenta</i>	color of URL links
verbose	<i>false</i>	be chatty
vtex		use VTeX backend
xetex		use XeTeX 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[options]{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 (unless the command is used in the argument of another command).

The optional argument *options* recognizes the hyperref options `pdfremotestartview`, `pdfnewwindow` and the following key value options:

**page:** Specifies the start page number of remote PDF documents. First page is 1.

**ismap:** Boolean key, if set to `true`, the URL should appended by the coordinates as query parameters by the PDF viewer.

**nextactionraw:** The value of key `/Next` of action dictionaries, see PDF specification.

`\url{URL}`

Similar to `\href{URL}{\nolinkurl{URL}}`. Depending on the driver `\href` also tries to detect the link type. Thus the result can be a url link, file link, ...

`\nolinkurl{URL}`

Write *URL* in the same way as `\url`, 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}{text}`

The link to the image referenced by the URL is inserted, using *text* as the anchor.

For drivers that produce HTML, the image itself is inserted by the browser, with the *text* being ignored completely.

`\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*

`\hyperref[label]{text}`

*text* is made into a link to the same place as `\ref{label}` would be linked.

`\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.

`\phantomsection`

This sets an anchor at this location. It works similar to `\hypertarget{}{}` with an automatically chosen anchor name. Often it is used in conjunction with `\addcontentsline` for sectionlike things (index, bibliography, preface). `\addcontentsline` refers to the latest previous location where an anchor is set. Example:

```

\cleardoublepage
\phantomsection
\addcontentsline{toc}{chapter}{\indexname}
\printindex

```

Now the entry in the table of contents (and bookmarks) for the index points to the start of the index page, not to a location before this page.

`\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 (re)defined in documents using `\(re)newcommand`; 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`.

<i>Macro</i>	<i>Default</i>
<code>\figurename</code>	Figure
<code>\tablename</code>	Table
<code>\partname</code>	Part
<code>\appendixname</code>	Appendix
<code>\equationname</code>	Equation
<code>\Itemname</code>	item
<code>\chaptername</code>	chapter
<code>\sectionname</code>	section
<code>\subsectionname</code>	subsection
<code>\subsubsectionname</code>	subsubsection
<code>\paragraphname</code>	paragraph
<code>\Hfootnotename</code>	footnote

<code>\AMSname</code>	Equation
<code>\theoremname</code>	Theorem
<code>\page</code>	page

Example for a redefinition if `babel` is used:

```
\usepackage[ngerman]{babel}
\addto\extrasngerman{%
  \def\subsectionautorefname{Unterkapitel}%
}
```

Hint: `\autoref` works via the counter name that the reference is based on. Sometimes `\autoref` chooses the wrong name, if the counter is used for different things. For example, it happens with `\newtheorem` if a lemma shares a counter with theorems. Then package `aliascnt` provides a method to generate a simulated second counter that allows the differentiation between theorems and lemmas:

```
\documentclass{article}

\usepackage{aliascnt}
\usepackage{hyperref}

\newtheorem{theorem}{Theorem}

\newaliascnt{lemma}{theorem}
\newtheorem{lemma}[lemma]{Lemma}
\aliascntresetthe{lemma}

\providecommand*{\lemmaautorefname}{Lemma}

\begin{document}

We will use \autoref{a} to prove \autoref{b}.

\begin{lemma}\label{a}
  Nobody knows.
\end{lemma}

\begin{theorem}\label{b}
  Nobody is right.
\end{theorem}.

\end{document}
```

`\autopageref{label}`

It replaces `\pageref` and adds the name for page in front of the page reference. First `\pageautorefname` is checked before `\pagename`.

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

`\ref*{label}`

`\pageref*{label}`

`\autoref*{label}`

`\autopageref*{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 “octal sequence commands” (`\345`): `\edef\3{\string\3}`
- Special glyphs of T<sub>E</sub>X: `\{`, `\%`, `\&`, `\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.
- L<sup>A</sup>T<sub>E</sub>X’s 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 EuroT<sub>E</sub>X paper distributed with `hyperref`.

`\begin{NoHyper}...\end{NoHyper}`

Sometimes we just don’t want the wretched package interfering with us. Define an environment we can put in manually, or include in a style file, which stops the hypertext functions doing anything. This is used, for instance, in the Elsevier classes, to stop `hyperref` playing havoc in the front matter.

## 4.1 Bookmark macros

### 4.1.1 Setting bookmarks

Usually `hyperref` automatically adds bookmarks for `\section` and similar macros. But they can also set manually.

```
\pdfbookmark[level]{text}{name}
```

creates a bookmark with the specified text and at the given level (default is 0). As name for the internal anchor name is used (in conjunction with level). Therefore the name must be unique (similar to `\label`).

```
\currentpdfbookmark{text}{name}
```

creates a bookmark at the current level.

```
\subpdfbookmark{text}{name}
```

creates a bookmark one step down in the bookmark hierarchy. Internally the current level is increased by one.

```
\belowpdfbookmark{text}{name}
```

creates a bookmark below the current bookmark level. However after the command the current bookmark level has not changed.

**Hint:** Package `bookmark` replaces `hyperref`'s bookmark organization by a new algorithm:

- Usually only one  $\LaTeX$  run is needed.
- More control over the bookmark appearance (color, font).
- Different bookmark actions are supported (external file links, URLs, ...).

Therefore I recommend using this package.

### 4.1.2 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\textttwosuperior\ + b\textttwosuperior\ =
    c\textttwosuperior
  }%
}
\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
}

```

## 4.2 Utility macros

`\hypercalsbp{dimen specification}`

`\hypercalsbp` takes a  $\TeX$  *dimen* specification and converts it to bp and returns the number without the unit. This is useful for options `pdfview`, `pdfstartview` and `pdfremotestartview`. Example:

```

\hypersetup{
  pdfstartview={FitBH \hypercalsbp{\paperheight-\topmargin-1in
  -\headheight-\headsep}
}

```

The origin of the PDF coordinate system is the lower left corner.

Note, for calculations you need either package `calc` or  $\varepsilon\text{-}\TeX$ . Nowadays the latter should automatically be enabled for  $\LaTeX$  formats. Users without  $\varepsilon\text{-}\TeX$ , please, look in the source documentation `hyperref.dtx` for further limitations.

Also `\hypercalsbp` cannot be used in option specifications of `\documentclass` and `\usepackage`, because  $\LaTeX$  expands the option lists of these commands. However package `hyperref` is not yet loaded and an undefined control sequence error would arise.

## 5 New Features<sup>3</sup>

### 5.1 Option ‘pdfinkmargin’

Option ‘pdfinkmargin’ is an experimental option for specifying a link margin, if the driver supports this. Default is 1 pt for supporting drivers.

- pdfTeX**
- The link area also depends on the surrounding box.
  - Settings have local effect.
  - When a page is shipped out, pdfTeX uses the current setting of the link margin for all links on the page.

- pdfmark**
- Settings have global effect.

---

<sup>3</sup>This section moved from the README file, needs more integration into the manual

- xetex**
- Settings must be done in the preamble or the first page and then have global effect. The key inserts the new (x)dvipdfmx special `\special{dvipdfmx:config g #1}` (with the unit removed).

**Other drivers** Unsupported.

## 5.2 Field option ‘calculatesortkey’

Fields with calculated values are calculated in document order by default. If calculated field values depend on other calculated fields that appear later in the document, then the correct calculation order can be specified with option ‘calculatesortkey’. Its value is used as key to lexicographically sort the calculated fields. The sort key do not need to be unique. Fields that share the same key are sorted in document order.

Currently the field option ‘calculatesortkey’ is only supported by the driver for pdfTeX.

## 5.3 Option ‘localanchorname’

When an anchor is set (e.g. via `\refstepcounter`, then the anchor name is globally set to the current anchor name.

For example:

```
\section{Foobar}
\begin{equation}\end{equation}
\label{sec:foobar}
```

With the default global setting (`localanchorname=false`) a reference to ‘sec:foobar’ jumps to the equation before. With option ‘localanchorname’ the anchor of the equation is forgotten after the environment and the reference ‘sec:foobar’ jumps to the section title.

Option ‘localanchorname’ is an experimental option, there might be situations, where the anchor name is not available as expected.

## 5.4 Option ‘customdriver’

The value of option ‘customdriver’ is the name of an external driver file without extension ‘.def’. The file must have `\ProvidesFile` with a version date and number that match the date and number of ‘hyperref’, otherwise a warning is given.

Because the interface, what needs to be defined in the driver, is not well defined and quite messy, the option is mainly intended to ease developing, testing, debugging the driver part.

## 5.5 Option ‘psdextra’

LaTeX’s NFSS is used to assist the conversion of arbitrary TeX strings to PDF strings (bookmarks, PDF information entries). Many math command names (`\geq`, `\notin`, ...) are not in control of NFSS, therefore they are defined with prefix ‘text’ (`\textgeq`, `\textnotin`, ...). They can be mapped to short names during the processing to PDF strings. The disadvantage is that they are many hundreds macros that need to be redefined for each PDF string conversion. Therefore this can be enabled or disabled as option ‘psdextra’. On default the option is turned off (set to ‘false’). Turning the option on means that the short names are available. Then `\geq` can directly be used instead of `\textgeq`.

## 5.6 `\XeTeXLinkBox`

When XeTeX generates a link annotation, it does not look at the boxes (as the other drivers), but only at the character glyphs. If there are no glyphs (images, rules, ...), then it does not generate a link annotation. Macro `\XeTeXLinkBox` puts its argument in a box and adds spaces at the lower left and upper right corners. An additional margin can be specified by setting it to the dimension register `\XeTeXLinkMargin`. The default is 2pt.

Example:

```
% xelatex
\documentclass{article}
\usepackage{hyperref}
\setlength{\XeTeXLinkMargin}{1pt}
\begin{document}
\section{Hello World}
\newpage
\label{sec:hello}
\hyperref[sec:hello]{%
  \XeTeXLinkBox{\rule{10mm}{10mm}}%
}
\end{document}
```

## 5.7 `\IfHyperBooleanExists` and `\IfHyperBoolean`

`\IfHyperBooleanExists{OPTION}{YES}{NO}`

If a hyperref `OPTION` is a boolean, that means it takes values ‘true’ or ‘false’, then `\IfHyperBooleanExists` calls YES, otherwise NO.

`\IfHyperBoolean{OPTION}{YES}{NO}`

Macro `\IfHyperBoolean` calls YES, if `OPTION` exists as boolean and is enabled. Otherwise NO is executed.

Both macros are expandable. Additionally option ‘stoppedearly’ is available. It is enabled if `\MaybeStopEarly` or `\MaybeStopNow` end hyperref prematurely.

## 5.8 `\unichar`

If a Unicode character is not supported by `puenc.def`, it can be given by using `\unichar`. Its name and syntax is inherited from package ‘ucs’. However it is defined independently for use in hyperref’s `\pdfstringdef` (that converts arbitrary TeX code to PDF strings or tries to do this).

Macro `\unichar` takes a TeX number as argument, examples for U+263A (WHITE SMILING FACE):

```
\unichar{"263A}% hexadecimal notation
\unichar{9786}% decimal notation
```

“” must not be a babel shorthand character or otherwise active. Otherwise prefix it with `\string`:

```
\unichar{\string"263A}% converts `"' to `"' with catcode 12 (other)
```

Users of (n)german packages or babel options may use `\dq` instead:

```
\unichar{\dq 263A}% \dq is double quote with catcode 12 (other)
```



## 5.9 `\ifpdfstringunicode`

Some features of the PDF specification needs PDF strings. Examples are bookmarks or the entries in the information dictionary. The PDF specification allows two encodings ‘PDFDocEncoding’ (8-bit encoding) and ‘Unicode’ (UTF-16). The user can help using `\texorpdfstring` to replace complicate TeX constructs by a representation for the PDF string. However `\texorpdfstring` does not distinguish the two encodings. This gap closes `\ifpdfstringunicode`. It is only allowed in the second argument of `\texorpdfstring` and takes two arguments, the first allows the full range of Unicode. The second is limited to the characters available in PDFDocEncoding.

As example we take a macro definition for the Vietnamese name of Han The Thanh. Correctly written it needs some accented characters, one character even with a double accent. Class ‘tugboat.cls’ defines a macro for the typesetted name:

```
\def\Thanh{%
  H\`an~%
  Th\^e\llap{\raise 0.5ex\hbox{\'{}}}}%
  ~Th\`anh%
}
```

It’s not entirely correct, the second accent over the ‘e’ is not an acute, but a hook. However standard LaTeX does not provide such an accent.

Now we can extend the definition to support hyperref. The first and the last word are already supported automatically. Characters with two or more accents are a difficult business in LaTeX, because the NFSS2 macros of the LaTeX kernel do not support more than one accent. Therefore also `puenc.def` misses support for them. But we can provide it using `\unichar`. The character in question is:

```
% U+1EC3 LATIN SMALL LETTER E WITH CIRCUMFLEX AND HOOK ABOVE
```

Thus we can put this together:

```
\def\Thanh{%
  H\`an~%
  \texorpdfstring{Th\^e\llap{\raise 0.5ex\hbox{\'{}}}}}%
  {\ifpdfstringunicode{Th\unichar{"1EC3}}{Th\^e}}}%
  ~Th\`anh%
}
```

For PDFDocEncoding (PD1) the variant above has dropped the second accent. Alternatively we could provide a representation without accents instead of wrong accents:

```
\def\Thanh{%
  \texorpdfstring{%
    H\`an~%
    Th\^e\llap{\raise 0.5ex\hbox{\'{}}}}}%
  ~Th\`anh%
  }{%
    \ifpdfstringunicode{%
      H\`an Th\unichar{"1EC3} Th\`anh%
    }{%
      Han The Thanh%
    }%
  }%
}
```

## 5.10 Customizing index style file with `\nohyperpage`

Since version 2008/08/14 v6.78f.

For hyperlink support in the index, `hyperref` inserts `\hyperpage` into the index macros. After processing with `Makeindex`, `\hyperpage` analyzes its argument to detect page ranges and page comma lists. However, only the standard settings are supported directly:

```
delim_r "--"
delim_n ", "
```

(See manual page/documentation of `Makeindex` that explains the keys that can be used in style files for `Makeindex`.) Customized versions of `delim_r`, `delim_n`, `suffix_2p`, `suffix_3p`, `suffix_mp` needs markup that `\hyperpage` can detect and knows that this stuff does not belong to a page number. Makro `\nohyperpage` serves as this markup. Put the customized code for these keys inside `\nohyperpage`, e.g.:

```
suffix_2p "\\nohyperpage{f.}"
suffix_3p "\\nohyperpage{ff.}"
```

(Depending on the typesetting tradition some space `"\"`, or `"~`" should be put before the first `f` inside `\nohyperpage`.)

## 5.11 Experimental option ‘ocgcolorlinks’

The idea are colored links, when viewed, but printed without colors. This new experimental option ‘ocgcolorlinks’ uses Optional Content Groups, a feature introduced in PDF 1.5.

- The option must be given for package loading: `\usepackage[ocgcolorlinks]{hyperref}`
- Main disadvantage: Links cannot be broken across lines. PDF reference 1.7: 4.10.2 “Making Graphical Content Optional”: Graphics state operations, such as setting the color, ..., are still applied. Therefore the link text is put in a box and set twice, with and without color.
- The feature can be switched of by `\hypersetup{ocgcolorlinks=false}` inside the document.
- Supported drivers: `pdftex`, `dvipdfm`
- The PDF version should be at least 1.5. It is automatically set for `pdfTeX`. Users of `dvipdfmx` set the version on the command line: `dvipdfmx -V 5`

## 5.12 Option ‘pdfa’

The new option ‘pdfa’ tries to avoid violations of PDF/A in code generated by `hyperref`. However, the result is usually not in PDF/A, because many features aren’t controlled by `hyperref` (XMP metadata, fonts, colors, driver dependend low level stuff, ...).

Currently, option ‘pdfa’ sets and disables the following items:

- Enabled annotation flags: Print, NoZoom, NoRotate [PDF/A 6.5.3].
- Disabled annotation flags: Hidden, Invisible, NoView [PDF/A 6.5.3].
- Disabled: Launch action ([PDF/A 6.6.1].
- Restricted: Named actions (NextPage, PrevPage, FirstPage, LastPage) [PDF/A 6.6.1].
- Many things are disabled in PDF formulars:
  - JavaScript actions [PDF/A 6.6.1]

- Trigger events (additional actions) [PDF/A 6.6.2]
- Push button (because of JavaScript)
- Interactive Forms: Flag NeedAppearances is the default ‘false’ (Because of this, hyperref’s implementation of Forms looks ugly). [PDF/A 6.9]

The default value of the new option ‘pdfa’ is ‘false’. It influences the loading of the package and cannot be changed after hyperref is loaded (`\usepackage{hyperref}`).

### 5.13 Option ‘linktoc’ added

The new option ‘linktoc’ allows more control which part of an entry in the table of contents is made into a link:

- ‘linktoc=none’ (no links)
- ‘linktoc=section’ (default behaviour, same as ‘linktocpage=false’)
- ‘linktoc=page’ (same as ‘linktocpage=true’)
- ‘linktoc=all’ (both the section and page part are links)

### 5.14 Option ‘pdfnewwindow’ changed

Before 6.77b:

- pdfnewwindow=true -> /NewWindow true
- pdfnewwindow=false -> (absent)
- unused pdfnewwindow -> (absent)

Since 6.77b:

- pdfnewwindow=true -> /NewWindow true
- pdfnewwindow=false -> /NewWindow false
- pdfnewwindow= -> (absent)
- unused pdfnewwindow -> (absent)

Rationale: There is a difference between setting to ‘false’ and an absent entry. In the former case the new document replaces the old one, in the latter case the PDF viewer application should respect the user preference.

### 5.15 Flag options for PDF forms

PDF form field macros (`\TextField`, `\CheckBox`, ...) support boolean flag options. The option name is the lowercase version of the names in the PDF specification (1.7):

[http://www.adobe.com/devnet/pdf/pdf\\_reference.html](http://www.adobe.com/devnet/pdf/pdf_reference.html)

[http://www.adobe.com/devnet/acrobat/pdfs/pdf\\_reference.pdf](http://www.adobe.com/devnet/acrobat/pdfs/pdf_reference.pdf)

Options (convert to lowercase) except flags in square brackets:

- Table 8.16 Annotation flags (page 608):
  - 1 Invisible
  - 2 Hidden (PDF 1.2)
  - 3 Print (PDF 1.2)
  - 4 NoZoom (PDF 1.3)
  - 5 NoRotate (PDF 1.3)
  - 6 NoView (PDF 1.3)
  - [7 ReadOnly (PDF 1.3)] ignored for widget annotations, see table 8.70
  - 8 Locked (PDF 1.4)
  - 9 ToggleNoView (PDF 1.5)
  - 10 LockedContents (PDF 1.7)
- Table 8.70 Field flags common to all field types (page 676):
  - 1 ReadOnly
  - 2 Required
  - 3 NoExport
- Table 8.75 Field flags specific to button fields (page 686):
  - 15 NoToggleToOff (Radio buttons only)
  - 16 Radio (set: radio buttons, clear: check box, pushbutton: clear)
  - 17 Pushbutton
  - 26 RadiosInUniso (PDF 1.5)
- Table 8.77 Field flags specific to text fields (page 691):
  - 13 Multiline
  - 14 Password
  - 21 FileSelect (PDF 1.4)
  - 23 DoNotSpellCheck (PDF 1.4)
  - 24 DoNotScroll (PDF 1.4)
  - 25 Comb (PDF 1.5)
  - 26 RichText (PDF 1.5)
- Table 8.79 Field flags specific to choice fields (page 693):
  - 18 Combo (set: combo box, clear: list box)
  - 19 Edit (only useful if Combo is set)
  - 20 (Sort) for authoring tools, not PDF viewers
  - 22 MultiSelect (PDF 1.4)
  - 23 DoNotSpellCheck (PDF 1.4) (only useful if Combo and Edit are set)
  - 27 CommitOnSelChange (PDF 1.5)

- Table 8.86 Flags for submit-form actions (page 704):
  - [1 Include/Exclude] unsupported, use ‘noexport’ (table 8.70) instead
  - 2 IncludeNoValueFields
  - [3 ExportFormat] handled by option ‘export’
  - 4 GetMethod
  - 5 SubmitCoordinates
  - [6 XFDF (PDF 1.4)] handled by option ‘export’
  - 7 IncludeAppendSaves (PDF 1.4)
  - 8 IncludeAnnotations (PDF 1.4)
  - [9 SubmitPDF (PDF 1.4)] handled by option ‘export’
  - 10 CanonicalFormat (PDF 1.4)
  - 11 ExclNonUserAnnots (PDF 1.4)
  - 12 ExclFKey (PDF 1.4)
  - 14 EmbedForm (PDF 1.5)

New option ‘export’ sets the export format of a submit action. Valid values are (upper- or lowercase):

- FDF
- HTML
- XFDF
- PDF (not supported by Acrobat Reader)

### 5.16 Option ‘pdfversion’

This is an experimental option. It notifies ‘hyperref’ about the intended PDF version. Currently this is used in code for PDF forms (implementation notes 116 and 122 of PDF spec 1.7).

Values: 1.2, 1.3, 1.4, 1.5, 1.6, 1.7. Values below 1.2 are not supported, because most drivers expect higher PDF versions.

The option must be used early, not after `\usepackage{hyperref}`.

In theory this option should also set the PDF version, but this is not generally supported.

- pdfTeX below 1.10a: unsupported. pdfTeX  $\geq$  1.10a and  $<$  1.30: `\pdfoptionpdfminorversion`  
pdfTeX  $\geq$  1.30: `\pdfminorversion`
- dvipdfm: configuration file, example: TeX Live 2007, `texmf/dvipdfm/config/config`, entry ‘V 2’.
- dvipdfmx: configuration file, example: TeX Live 2007, `texmf/dvipdfm/dvipdfmx.cfg`, entry ‘V 4’.
- Ghostscript: option `-dCompatibilityLevel` (this is set in ‘ps2pdf12’, ‘ps2pdf13’, ‘ps2pdf14’).

The current PDF version is used as default if this version can be detected (only pdfTeX  $\geq$  1.10a). Otherwise the lowest version 1.2 is assumed. Thus ‘hyperref’ tries to avoid PDF code that breaks this version, but is free to use ignorable higher PDF features.

### 5.17 Field option ‘name’

Many form objects uses the label argument for several purposes:

- Laid out label.
- As name in HTML structures.

Code that is suitable for layouting with TeX can break in the structures of the output format. If option ‘name’ is given, then its value is used as name in the different output structures. Thus the value should consist of letters only.

### 5.18 Option ‘pdfencoding’

The PDF format allows two encodings for bookmarks and entries in the information dictionary: PDFDocEncoding and Unicode as UTF-16BE. Option “pdfencoding” selects between these encodings:

- “pdfdoc” uses PDFDocEncoding. It uses just one byte per character, but the supported characters are limited (244 in PDF-1.7).
- “unicode” sets Unicode. It is encoded as UTF-16BE. Two bytes are used for most characters, surrogates need four bytes.
- “auto” PDFDocEncoding if the string does not contain characters outside the encoding and Unicode otherwise.

### 5.19 Color options/package hycolor

See documentation of package ‘hycolor’.

### 5.20 Option pdfusetitle

If option pdfusetitle is set then hyperref tries to derive the values for pdftitle and pdfauthor from `\title` and `\author`. An optional argument for `\title` and `\author` is supported (class `amsart`).

### 5.21 Starred form of `\autoref`

`\autoref*` generates a reference without link as `\ref*` or `\pageref*`.

### 5.22 Link border style

Links can be underlined instead of the default rectangle or options “colorlinks”, “frenchlinks”. This is done by option `pdfborderstyle={/S/U/W 1}`

Some remarks:

- AR7/Linux seems to have a bug, that don’t use the default value “1” for the width, but zero, thus that the underline is not visible without “/W 1”. The same applies for dashed boxes, eg.: `pdfborderstyle=/S/D/D[3 2]/W 1`
- The syntax is described in the PDF specification, look for “border style”, eg. Table 8.13 “Entries in a border style dictionary” (specification for version 1.6)
- The border style is removed by `pdfborderstyle=` This is automatically done if option `colorlinks` is enabled.

- Be aware that not all PDF viewers support this feature, not even Acrobat Reader itself:

Some support:

- AR7/Linux: "underline" and "dashed", but the border width must be given.
- xpdf 3.00: "underline" and "dashed"

Unsupported:

- AR5/Linux
- ghostscript 8.50

### 5.23 Option "bookmarksdepth"

The depth of the bookmarks can be controlled by the new option "bookmarksdepth". The option acts globally and distinguishes three cases:

- "bookmarksdepth" without value Then hyperref uses the current value of counter "tocdepth". This is the compatible behaviour and the default.
- "bookmarksdepth=<number>", the value is number (also negative): The depth for the bookmarks are set to this number.
- "bookmarksdepth=<name>" The <name> is a document division name (part, chapter, ...). It must not start with a digit or minus to avoid mixing up with the number case. Internally hyperref uses the value of macro "\toclevel@<name>". Examples:

```
\hypersetup{bookmarksdepth=paragraph}
\hypersetup{bookmarksdepth=4} % same as before
\hypersetup{bookmarksdepth} % counter "tocdepth" is used
```

### 5.24 Option "pdfescapeform"

There are many places where arbitrary strings end up as PS or PDF strings. The PS/PDF strings in parentheses form require the protection of some characters, e.g. unmatched left or right parentheses need escaping or the escape character itself (backslash). Since 2006/02/12 v6.75a the PS/PDF driver should do this automatically. However I assume a problem with compatibility, especially regarding the form part where larger amounts of JavaScript code can be present. It would be a pain to remove all the escaping, because an additional escaping layer can falsify the code.

Therefore a new option pdfescapeform was introduced:

- pdfescapeform=false Escaping for the formulars are disabled, this is the compatibility behaviour, therefore this is the default.
- pdfescapeform=true Then the PS/PDF drivers do all the necessary escaping. This is the logical choice and the recommended setting. For example, the user writes JavaScript as JavaScript and do not care about escaping characters for PS/PDF output.

### 5.25 Default driver setting

(hyperref >= 6.72s) If no driver is given, hyperref tries its best to guess the most suitable driver. Thus it loads "hpdftex", if pdfTeX is detected running in PDF mode. Or it loads the corresponding VTeX driver for VTeX's working modes. Unhappily many driver programs run after the TeX compiler, so hyperref does not have a chance (dvips, dviplfm, ...). In this case driver "hypertex" is loaded that supports the HyperTeX features that are recognized by xdvi for example. This behaviour, however, can easily be changed in the configuration file "hyperref.cfg":

```
\providecommand*{\Hy@defaultdriver}{hdvips}
```

for dvips, or

```
\providecommand*{\Hy@defaultdriver}{hypertex}
```

for the default behaviour of hyperref.

See also the new option ‘driverfallback’.

## 5.26 Backref entries

Alternative interface for formatting of backref entries, example:

```
\documentclass[12pt,UKenglish]{article}

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

% Some language options are detected by package backref.
% This affects the following macros:
% \backrefpagesname
% \backrefsectionsname
% \backrefsep
% \backreftwosep
% \backreflastsep

\renewcommand*{\backref}[1]{
  % default interface
  % #1: backref list
  %
  % We want to use the alternative interface,
  % therefore the definition is empty here.
}
\renewcommand*{\backrefalt}[4]{%
  % alternative interface
  % #1: number of distinct back references
  % #2: backref list with distinct entries
  % #3: number of back references including duplicates
  % #4: backref list including duplicates
  \par
  #3 citation(s) on #1 page(s): #2,\par
  \ifnum#1=1 %
    \ifnum#3=1 %
      1 citation on page %
    \else
      #3 citations on page %
    \fi
  \else
    #3 citations on #1 pages %
  \fi
  #2,\par
  \ifnum#3=1 %
    1 citation located at page %
```



```

\else
  #3 citations located at pages %
\fi
#4.\par
}

% The list of distinct entries can be further refined:
\renewcommand*{\backrefentrycount}[2]{%
  % #1: the original backref entry
  % #2: the count of citations of this entry,
  %    in case of duplicates greater than one
  #1%
  \ifnum#2>1 %
    ~(#2)%
  \fi
}

\begin{document}

\section{Hello}
\cite{ref1, ref2, ref3, ref4}
\section{World}
\cite{ref1, ref3}
\newpage

\section{Next section}
\cite{ref1}
\newpage

\section{Last section}
\cite{ref1, ref2}
\newpage

\pdfbookmark[1]{Bibliography}{bib}
\begin{thebibliography}{99}
\bibitem{ref1} Dummy entry one.

\    \bibitem{ref2} Dummy entry two.

\    \bibitem{ref3} Dummy entry three.

\    \bibitem{ref4} Dummy entry four.

\end{thebibliography}

\end{document}

```

### 5.27 \phantomsection

Set an anchor at this location. It is often used in conjunction with `\addcontentsline` for sectionlike things (index, bibliography, preface). `\addcontentsline` refers to the latest previous location where an anchor is set.

```

\cleardoublepage
\phantomsection
\addcontentsline{toc}{chapter}{\indexname}
\printindex

```

Now the entry in the table of contents (and bookmarks) for the index points to the start of the index page, not to a location before this page.

## 6 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

## 7 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}
```

```
\LayoutCheckField{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\DefaultHeightofTextMultiline{4\baselineskip}
\def\DefaultWidthofText{3cm}
```

## 7.1 Forms environment parameters

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

## 7.2 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`)

<code>accesskey</code>	<i>key</i>	(as per HTML)
<code>align</code>	<i>number</i> <i>0</i>	alignment within text field; 0 is left-aligned, 1 is centered, 2 is right-aligned.
<code>altname</code>	<i>name</i>	alternative name, the name shown in the user interface
<code>backgroundcolor</code>		color of box
<code>bordercolor</code>		color of border

bordersep			box border gap
borderwidth		<i>1</i>	width of box border, the value is a dimension or a number with default unit bp
calculate			JavaScript code to calculate the value of the field
charsize	dimen		font size of field text
checkboxsymbol	char	<i>4 (✓)</i>	symbol used for check boxes (ZapfDingbats), the value is a character or <code>\ding{number}</code> , see package <code>pifont</code> from bundle <code>psnfss</code>
checked	boolean	<i>false</i>	whether option selected by default
color			color of text in box
combo	boolean	<i>false</i>	choice list is 'combo' style
default			default value
disabled	boolean	<i>false</i>	field disabled
format			JavaScript code to format the field
height	dimen		height of field box
hidden	boolean	<i>false</i>	field hidden
keystroke			JavaScript code to control the keystrokes on entry
mappingname	name		the mapping name to be used when exporting the field data
maxlen	number	<i>0</i>	number of characters allowed in text field
menulength	number	<i>4</i>	number of elements shown in list
multiline	boolean	<i>false</i>	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	<i>false</i>	text field is 'password' style
popdown	boolean	<i>false</i>	choice list is 'popdown' style
radio	boolean	<i>false</i>	choice list is 'radio' style
radiosymbol	char	<i>H (★)</i>	symbol used for radio fields (ZapfDingbats), the value is a character or <code>\ding{number}</code> , see package <code>pifont</code> from bundle <code>psnfss</code>
readonly	boolean	<i>false</i>	field is readonly
rotation	number	<i>0</i>	rotation of the widget annotation (degree, counterclockwise, multiple of 90)
tabkey			(as per HTML)
validate			JavaScript code to validate the entry
value			initial value
width	dimen		width of field box

## 8 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
```

## 9 Special support for other packages

Package `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 L<sup>A</sup>T<sub>E</sub>X's normal `\label/\ref` mechanisms and per-chapter tables of contents, respectively.

### 9.1 Package Compatibility

Currently only package loading orders are available:

Note: `hyperref` loads package "nameref" at `\begin{document}`. Sometimes this is too late, thus this package must be loaded earlier.

#### 9.1.1 algorithm

```
\usepackage{float}
\usepackage{hyperref}
\usepackage[chapter]{algorithm}% eg.
```

### 9.1.2 amsmath

The environments `equation` and `eqnarray` are not supported too well. For example, there might be spacing problems (`eqnarray` isn't recommended anyway, see [CTAN:info/l2tabu/](http://CTAN:info/l2tabu/), the situation for `equation` is unclear, because nobody is interested in investigating). Consider using the environments that package `amsmath` provide, e.g. `gather` for `equation`. The environment `equation` can even be redefined to use `gather`:

```
\usepackage{amsmath}
\let\equation\gather
\let\endequation\endgather
```

### 9.1.3 amsrefs

Package loading order:

```
\usepackage{hyperref}
\usepackage{amsrefs}
```

### 9.1.4 arydshln, longtable

Package `longtable` must be put before `hyperref` and `arydshln`, `hyperref` after `arydshln` generates an error, thus the resulting package order is then:

```
\usepackage{longtable}
\usepackage{hyperref}
\usepackage{arydshln}
```

### 9.1.5 babel/magyar.ldf

The old version 2005/03/30 v1.4j will not work. You need at least version 1.5, maintained by Péter Szabó, see [CTAN:language/hungarian/babel/](http://CTAN:language/hungarian/babel/).

### 9.1.6 babel/spanish.ldf

Babel's `spanish.ldf` redefines `\.` to support `\...`. In bookmarks (`\pdfstringdef`) only `\.` is supported. If `\...` is needed, `\texorpdfstring{\...}{\dots}` can be used instead.

### 9.1.7 bibentry

Workaround:

```
\makeatletter
\let\saved@bibitem\@bibitem
\makeatother

\usepackage{bibentry}
\usepackage{hyperref}

\begin{document}

\begingroup
  \makeatletter
  \let\@bibitem\saved@bibitem
  \nobibliography{database}
\endgroup
```

**9.1.8 bigfoot**

Hyperref does not support package ‘bigfoot’. And package ‘bigfoot’ does not support hyperref’s footnotes and disables them (`hyperfootnotes=false`).

**9.1.9 chappg**

Package ‘chappg’ uses `\@addtoreset` that is redefined by ‘hyperref’. The package order is therefore:

```
\usepackage{hyperref}
\usepackage{chappg}
```

**9.1.10 cite**

This is from Mike Shell: `cite.sty` cannot currently be used with `hyperref`. However, I can do a workaround via:

```
\makeatletter
\def\NAT@parse{\typeout{This is a fake Natbib command to fool Hyperref.}}
\makeatother

\usepackage[hypertex]{hyperref}
```

so that `hyperref` will not redefine any of the `biblabel` stuff - so `cite.sty` will work as normal - although the citations will not be hyperlinked, of course (But this may not be an issue for many people).

**9.1.11 count1to**

Package ‘count1to’ adds several `\@addtoreset` commands that confuse ‘hyperref’. Therefore `\theH<...>` has to be fixed:

```
\usepackage{count1to}
\AtBeginDocument{% *after* \usepackage{count1to}
  \renewcommand*\theHsection{\theHchapter.\arabic{section}}%
  \renewcommand*\theHsubsection{\theHsection.\arabic{subsection}}%
  \renewcommand*\theHsubsubsection{\theHsubsection.\arabic{subsubsection}}%
  \renewcommand*\theHparagraph{\theHsubsubsection.\arabic{paragraph}}%
  \renewcommand*\theHsubparagraph{\theHparagraph.\arabic{subparagraph}}%
}
```

**9.1.12 dblacnt**

`pd1enc.def` or `puenc.def` should be loaded before:

```
\usepackage{hyperref}
\usepackage{dblacnt}
```

or see entry for “vietnam”.

**9.1.13 easyeqn**

Not compatible, breaks.



**9.1.14 ellipsis**

This packages redefines `\textellipsis` after package `hyperref` (`pd1enc.def/puenc.def` should be loaded before):

```
\usepackage{hyperref}
\usepackage{ellipsis}
```

**9.1.15 float**

```
\usepackage{float}
\usepackage{hyperref}
```

- Several `\caption` commands are not supported inside one float object.
- Anchor are set at top of the float object, if its style is controlled by `float.sty`.

**9.1.16 endnotes**

Unsupported.

**9.1.17 foiltex**

Update to version 2008/01/28 v2.1.4b: Since version 6.77a `hyperref` does not hack into `\@begindvi`, it uses package ‘`atbegshi`’ instead, that hooks into `\shipout`. Thus the patch of ‘`foils.cls`’ regarding `hyperref` is now obsolete and causes an undefined error message about `\@hyperfixhead`. This is fixed in `FoilTeX 2.1.4b`.

**9.1.18 footnote**

This package is not supported, you have to disable `hyperref`’s footnote support by using option “`hyperfootnotes=false`”.

**9.1.19 geometry**

Driver ‘`dvipdfm`’ and program ‘`dvipdfm`’ might generate a warning: Sorry. Too late to change page size Then prefer the program ‘`dvipdfmx`’ or use one of the following workarounds to move the `\special` of `geometry` to an earlier location:

```
\documentclass[dvipdfm]{article}% or other classes
\usepackage{atbegshi}
\AtBeginDocument{%
  \let\OrgAtBeginDvi\AtBeginDvi
  \let\AtBeginDvi\AtBeginShipoutFirst
}
\usepackage[
  paperwidth=170mm,
  paperheight=240mm
]{geometry}
\AtBeginDocument{%
  \let\AtBeginDvi\OrgAtBeginDvi
}
\usepackage{hyperref}
```

or

```

\documentclass[dvipdfm]{article}% or other classes
\usepackage{atbegshi}
\let\AtBeginDvi\AtBeginShipoutFirst
\usepackage[
  paperwidth=170mm,
  paperheight=240mm
]{geometry}
\usepackage{hyperref}

```

### 9.1.20 IEEEtran.cls

version  $\geq$  V1.6b (because of `\@makecaption`, see ChangeLog)

### 9.1.21 index

version  $\geq$  1995/09/28 v4.1 (because of `\addcontentsline` redefinition)

### 9.1.22 lastpage

Compatible.

### 9.1.23 linguex

```

\usepackage{hyperref}
\usepackage{linguex}

```

### 9.1.24 ltabptch

```

\usepackage{longtable}
\usepackage{ltabptch}
\usepackage{hyperref}

```

### 9.1.25 mathenv

Unsupported.

Both ‘mathenv’ and ‘hyperref’ messes around with environment ‘eqnarray’. You can load ‘mathenv’ after ‘hyperref’ to avoid an error message. But `\label` will not work inside environment ‘eqnarray’ properly, for example.

### 9.1.26 minitoc-hyper

This package is obsolete, use the uptodate original package minitoc instead.

### 9.1.27 multind

```

\usepackage{multind}
\usepackage{hyperref}

```

### 9.1.28 natbib

```

\usepackage{natbib}
\usepackage{hyperref}

```

**9.1.29 nomencl**

Example for introducing links for the page numbers:

```
\renewcommand*{\pagedeclaration}[1]{\unskip, \hyperpage{#1}}
```

For equations the following might work:

```
\renewcommand*{\eqdeclaration}[1]{%
  \hyperlink{equation.#1}{(Equation~#1)}%
}
```

But the mapping from the equation number to the anchor name is not available in general.

**9.1.30 parskip**

```
\usepackage{parskip}
\usepackage{hyperref}[2012/08/20]
```

Both packages want to redefine \@starttoc.

**9.1.31 prettyref**

%%% example for prettyref %%%

```
\documentclass{article}
\usepackage{prettyref}
\usepackage[pdftex]{hyperref}
```

```
%\newreformat{FIG}{Figure~\ref{#1}}% without hyperref
\newreformat{FIG}{\hyperref[#{#1}]{Figure~\ref*{#1}}}
```

```
\begin{document}
  This is a reference to \prettyref{FIG:ONE}.
  \newpage
  \begin{figure}
    \caption{This is my figure}
    \label{FIG:ONE}
  \end{figure}
\end{document}
%%% example for prettyref %%%
```

**9.1.32 ntheorem**

ntheorem-hyper.sty is an old patched version of ntheorem.sty.

Newer versions of ntheorem know the option hyperref:

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

But there are still unsolved problems (options thref, ...).

**9.1.33 setspace**

```
\usepackage{setspace}
\usepackage{hyperref}
```

**9.1.34 sidecap**

Before 2002/05/24 v1.5h:  
`\usepackage{nameref}`  
`\usepackage{hyperref}`  
`\usepackage{sidecap}`

**9.1.35 subfigure**

1995/03/06 v2.0:  
`\usepackage{subfigure}`  
`\usepackage{hyperref}`  
`% hypertexnames is set to false.`

v2.1:  
`\usepackage{nameref}`  
`\usepackage{subfigure}`  
`\usepackage{hyperref}`  
or  
`\usepackage{hyperref}`  
`\usepackage{subfigure}`

v2.1.2:  
please update

v2.1.3:  
`\usepackage{hyperref}`  
`\usepackage{subfigure}`  
or vice versa?

**9.1.36 titleref**

`\usepackage{nameref}`  
`\usepackage{titleref}% without usetoc`  
`\usepackage{hyperref}`

**9.1.37 tabularx**

Linked footnotes are not supported inside environment ‘tabularx’, because they uses the optional argument of `\footnotetext`, see section ‘Limitations’. Before version 2011/09/28 6.82i hyperref had disabled footnotes entirely by ‘hyperfootnotes=false’.

**9.1.38 titlesec**

”nameref” supports titlesec, but hyperref does not (unsolved is the anchor setting, missing with unnumbered section, perhaps problems with page breaks with numbered ones).

**9.1.39 ucs/utf8x.def**

The first time a multibyte UTF8 sequence is called, it does some calculations and stores the result in a macro for speeding up the next calls of that UTF8 sequence. However this makes the first call non-expandable and will break if used in information entries or bookmarks. Package ”ucs” offers `\PrerenderUnicode` or `\PreloadUnicodePage` to solve this:

`\usepackage{ucs}`  
`\usepackage[utf8x]{inputenc}`  
`\usepackage{hyperref}% or with option unicode`

```
\PrerenderUnicode{^^c3^^b6}% or \PrerenderUnicodePage{1}
\hypersetup{pdftitle={Umlaut example: ^^c3^^b6}}
```

The notation with two carets avoids trouble with 8-bit bytes for the README file, you can use the characters directly.

#### 9.1.40 varioref

There are too many problems with varioref. Nobody has time to sort them out. Therefore this package is now unsupported.

Perhaps you are lucky and some of the features of varioref works with the following loading order:

```
\usepackage{nameref}
\usepackage{varioref}
\usepackage{hyperref}
```

Also some babel versions can be problematic. For example, 2005/05/21 v3.8g contains a patch for varioref that breaks the hyperref support for varioref.

Also unsupported:

- `\Ref`, `\Vref` do not uppercase the first letter.
- `\vpageref[...]` On the same page a previous space is not suppressed.

#### 9.1.41 verse

Version 2005/08/22 v2.22 contains support for hyperref.

For older versions see example from `de.comp.text.tex` (2005/08/11, slightly modified):

```
\documentclass{article}

% package order does not matter
\usepackage{verse}
\usepackage{hyperref}

\makeatletter
% make unique poemline anchors
\newcounter{verse@env}
\setcounter{verse@env}{0}
\let\org@verse\verse
\def\verse{%
  \stepcounter{verse@env}%
  \org@verse
}
\def\theHpoemline{\arabic{verse@env}.\thepoemline}

% add anchor for before \addcontentsline in \@vsptitle
\let\org@vsptitle\@vsptitle
\def\@vsptitle{%
  \phantomsection
  \org@vsptitle
}
\makeatother
```

```

\begin{document}

\poemtitle{Poem 1}
\begin{verse}
An one-liner.
\end{verse}

\newpage

\poemtitle{Poem 2}
\begin{verse}
Another one-liner.
\end{verse}

\end{document}

```

#### 9.1.42 vietnam

```

% pdlenc.def should be loaded before package dblacnt:
\usepackage[PD1,OT1]{fontenc}
\usepackage{vietnam}
\usepackage{hyperref}

```

#### 9.1.43 XeTeX

Default for the encoding of bookmarks is ‘pdfencoding=auto’. That means the strings are always treated as unicode strings. Only if the string restricts to the printable ASCII set, it is written as ASCII string. The reason is that the `\special` does not support PDFDocEncoding.

XeTeX uses the program `xdvipdfmx` for PDF output generation. This program behaves a little different from `dvipdfm`, because of the supported Unicode characters. Strings for bookmarks or information entries can be output directly. The big chars (char code > 255) are written in UTF-8 and `xdvipdfmx` tries to convert them to UTF-16BE. However `hyperref` already provides PDF strings encoded in UTF-16BE, thus the result is a warning

```

"Failed to convert input string to UTF16..."

```

The best way would be, if `xdvipdfm` could detect the byte order marker (`\376\377`) and skips the conversion if that marker is present.

For the time being I added the following to `hyperref`, when option ‘pdfencoding=auto’ is set (default for XeTeX): The string is converted back to big characters thus that the string is written as UTF-8. But I am very unhappy with this solution. Main disadvantage: Two versions of `\pdfstringdef` are needed:

a) The string is converted back to big characters for the “tainted keys” of `xdvipdfmx` (`spc_pdfm.c: default_taintkeys`). The subset `hyperref` uses is `/Title`, `/Author`, `/Subject`, `/Keywords`, `/Creator`, `/Producer`, `/T`. Any changes of this set in `xdvipdfmx` cannot be detected by `hyperref`.

b) Without conversion for the other strings, providing UTF16be directly. Examples: Prefix of page labels, some elements of formulars.

Thus **each** application that uses `\pdfstringdef` now must check, if it defines a string for some of the tainted keys. If yes, then the call of `\pdfstringdef` should be preceded by `"\csname HyPsd@XeTeXBigCharstrue\endcsname"`. Example: package bookmark.

## 10 Limitations<sup>4</sup>

### 10.1 Wrapped/broken link support

Only few drivers support automatically wrapped/broken links, e.g. pdftex, dvi<sub>pdfm</sub>, hypertext. Other drivers lack this feature, e.g. dvips, dvipsone.

Workarounds:

- For long section or caption titles in the table of contents or list of figures/tables option "linktocpage" can be used. Then the page number will be a link, and the overlong section title is not forced into an one line link with overfull \hbox warning.
- "\url"s are caught by package "breakurl".
- The option "breaklinks" is intended for internal use. But it can be used to force link wrapping, e.g. when printing a document. However, when such a document is converted to PDF and viewed with a PDF viewer, the active link area will be misplaced.

Another limitation: some penalties are "optimized" by TeX, thus there are missing break points, especially within \url. (See thread "hyperref.sty, breaklinks and url.sty 3.2" in comp.text.tex 2005-09).

### 10.2 Links across pages

In general they have problems:

- Some driver doesn't support them at all (see above).
- The driver allows it, but the link result might include the footer and/or header, or an error message can occur sometimes.

### 10.3 Footnotes

LaTeX allows the separation of the footnote mark and the footnote text (`\footnotemark`, `\footnotetext`). This interface might be enough for visual typesetting. But the relation between `\footnotemark` to `\footnotetext` is not as strong as `\ref` to `\label`. Therefore it is not clear in general which `\footnotemark` references which `\footnotetext`. But that is necessary to implement hyperlinking. Thus the implementation of hyperref does not support the optional argument of `\footnotemark\verb` and `\footnotetext`.

## 11 Hints<sup>5</sup>

### 11.1 Spaces in option values

Unhappily LaTeX strips spaces from options if they are given in `\documentclass` or `\usepackage` (or `\RequirePackage`), e.g.:

```
\usepackage[pdfborder=0 0 1]{hyperref}
```

Package hyperref now gets

```
pdfborder=001
```

and the result is an invalid PDF file. As workaround braces can be used:

<sup>4</sup>This section moved from the README file, needs more integration into the manual

<sup>5</sup>This section moved from the README file, needs more integration into the manual

```
\usepackage[pdfborder={0 0 1}]{hyperref}
```

Some options can also be given in `\hypersetup`

```
\hypersetup{pdfborder=0 0 1}
```

In `\hypersetup` the options are directly processed as key value options (see package `keyval`) without space stripping in the value part.

Alternatively, LaTeX's option handling system can be adapted to key value options by one of the packages "kvoptions-patch" (from project "kvoptions") or "xkvltxp" (from project "xsetkeys").

## 11.2 Index with makeindex

- Package `hyperref` adds `\hyperpage` commands by the `encap` mechanism (see documentation of `Makeindex`), if option `hyperindex` is set (default). `\hyperpage` uses the page anchors that are set by `hyperref` at each page (default). However in the default case page numbers are used in anchor names in arabic form. If the page numbers in other formats are used (book class with `\frontmatter`, `\romannumbering`, ...), then the page anchors are not unique. Therefore option "plainpages=false" is recommended.
- The `encap` mechanism of `Makeindex` allows to use one command only (see documentation of `Makeindex`). If the user sets such a command, `hyperref` suppresses its `\hyperpage` command. With logical markup this situation can easily be solved:

```
\usepackage{makeidx}
\makeindex
\usepackage[hyperindex]{hyperref}
\newcommand*{\main}[1]{\textbf{\hyperpage{#1}}}
...
\index{Some example|main}
```

- Scientific Word/Scientific WorkPlace users can use package `robustindex` with `hyperindex=false`.
- Other `encap` characters can be set by option "encap". Example for use of "?":

```
\usepackage[encap=?]{hyperref}
```

- An other possibility is the insertion of `\hyperpage` by a style file for `makeindex`. For this case, `hyperref`'s insertion will be disabled by "hyperindex=false". `\hyperpage` will be defined regardless of setting of `hyperindex`.

```
%%% cut %%% hyperindex.ist %%% cut %%%
delim_0 ", \hyperpage{"
delim_1 ", \hyperpage{"
delim_2 ", \hyperpage{"
delim_n "}, \hyperpage{"
delim_t "}"
encap_prefix "}"
encap_infix "{\hyperpage{"
encap_suffix "}"
%%% cut %%% hyperindex.ist %%% cut %%%
```



### 11.3 Warning "bookmark level for unknown <foobar> defaults to 0"

Getting rid of it:

```
\makeatletter
\providecommand*\toclevel@<foobar>{0}
\makeatother
```

### 11.4 Link anchors in figures

The caption command increments the counter and here is the place where hyperref set the corresponding anchor. Unhappily the caption is set below the figure, so the figure is not visible if a link jumps to a figure. In this case, try package "hycap.sty" that implements a method to circumvent the problem.

### 11.5 Additional unicode characters in bookmarks and pdf information entries:

```
\documentclass[pdftex]{article}
\usepackage[unicode]{hyperref}
```

Support for additional unicode characters:

Example: `\{a}` and `\d{a}`

1. Get a list with unicode data, eg:

<http://www.unicode.org/Public/UNIDATA/UnicodeData.txt>

2. Identify the characters (`\{a}`, `\d{a}`):

```
0227;LATIN SMALL LETTER A WITH DOT ABOVE;...
```

```
1EA1;LATIN SMALL LETTER A WITH DOT BELOW;...
```

3. Calculate the octal code:

The first characters of the line in the file are hex values, convert each byte and prepend them with a backslash. (This will go into the PDF file.)

```
0227 -> \002\047
```

```
1EA1 -> \036\241
```

4. Transform into a form understood by hyperref:

Hyperref must know where the first byte starts, this is marked by "9" (8 and 9 cannot occur in octal numbers):

```
\002\047 -> \9002\047
```

```
\036\241 -> \9036\241
```

Optional: "8" is used for abbreviations:

```
\900 = \80, \901 = \81, \902 = \82, ...
```

```
\9002\047 -> \82\047
```

5. Declare the character with LaTeX:

```
\DeclareTextCompositeCommand{\}{PU}{a}{\82\047}
\DeclareTextCompositeCommand{\d}{PU}{a}{\9036\241}
```

```
\begin{document}
\section{\={a}, \d{a}, \'{a}, \.a}
\end{document}
```

## 11.6 Footnotes

The footnote support is rather limited. It is beyond the scope to use `\footnotemark` and `\footnotetext` out of order or reusing `\footnotemark`. Here you can either disable hyperref's footnote support by "hyperfootnotes=false" or fiddle with internal macros, nasty examples:

```
\documentclass{article}
\usepackage{hyperref}
\begin{document}
Hello%
\footnote{The first footnote}
World%
\addtocounter{footnote}{-1}%
\addtocounter{Hfootnote}{-1}%
\footnotemark.
\end{document}
```

or

```
\documentclass{article}

\usepackage{hyperref}

\begin{document}

\makeatletter

A%
\footnotemark
\let\saved@Href@A\Hy@footnote@currentHref
% remember link name
B%
\footnotemark
\let\saved@Href@B\Hy@footnote@currentHref
b%
\addtocounter{footnote}{-1}%
\addtocounter{Hfootnote}{-1}% generate the same anchor
\footnotemark
C%
\footnotemark
\let\saved@Href@C\Hy@footnote@currentHref

\addtocounter{footnote}{-2}%
\let\Hy@footnote@currentHref\saved@Href@A
\footnotetext{AAAA}%
\addtocounter{footnote}{1}%
\let\Hy@footnote@currentHref\saved@Href@B
\footnotetext{BBBBB}%
\addtocounter{footnote}{1}%
\let\Hy@footnote@currentHref\saved@Href@C
\footnotetext{CCCC}%

\end{document}
```

## 11.7 Subordinate counters

Some counters do not have unique values and require the value of other counters to be unique. For example, sections or figures might be numbered within chapters or `\newtheorem` is used with an optional counter argument. Internally LaTeX uses `\@addtoreset` to reset a counter in dependency to another counter. Package `hyperref` hooks into `\@addtoreset` to catch this situation. Also `\numberwithin` of package `amsmath` is caught by `hyperref`.

However, if the definition of subordinate counters take place before `hyperref` is loaded, the old meaning of `\@addtoreset` is called without `hyperref`'s additions. Then the companion counter macro `\theH<counter>` can be redefined accordingly. Or move the definition of subordinate counters after `hyperref` is loaded.

Example for `\newtheorem`, problematic case:

```
\newtheorem{corA}{CorollaryA}[section]
\usepackage{hyperref}
```

Solution a)

```
\usepackage{hyperref}
\newtheorem{corA}{CorollaryA}[section]
```

Solution b)

```
\newtheorem{corA}{CorollaryA}[section]
\usepackage{hyperref}
\newcommand*{\theHcorA}{\theHsection.\number\value{corA}}
```

## 12 History and acknowledgments

The original authors of `hyperbasics.tex` and `hypertex.sty`, from which this package descends, are Tanmoy Bhattacharya and Thorsten Ohl. Package `hyperref` started as a simple port of their work to  $\text{\LaTeX} 2_{\epsilon}$  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  $\text{\LaTeX}$ . 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 into the later versions including native PDF support uncovered a good many bugs, and his testing is appreciated. Hans Hagen provided a lot of insight into PDF.

Berthold Horn provided help, encouragement and sponsorship for the `dvipson` 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 Peter recreated the manual source in July 2003 after it had been lost.

Especially 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`.

## 13 GNU Free Documentation License

Version 1.2, November 2002

Copyright © 2000,2001,2002 Free Software Foundation, Inc.  
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

### Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

### 13.1 Applicability and definitions

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic

paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, L<sup>A</sup>T<sub>E</sub>X input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

## 13.2 Verbatim copying

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 13.3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

## 13.3 Copying in quantity

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

### 13.4 Modifications

You may copy and distribute a Modified Version of the Document under the conditions of sections 13.2 and 13.3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the “History” section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled “Acknowledgements” or “Dedications”, Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled “Endorsements”. Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled “Endorsements” or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section Entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

### 13.5 Combining documents

You may combine the Document with other documents released under this License, under the terms defined in section 13.4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements”.

### 13.6 Collections of documents

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

### 13.7 Aggregation with independent works

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 13.3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

### 13.8 Translation

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 13.4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 13.4) to Preserve its Title (section 13.1) will typically require changing the actual title.

### 13.9 Termination

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.



### 13.10 Future revisions of this license

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

### Addendum: how to use this license for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright © YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled “GNU Free Documentation License”.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.