

# The rerunfilecheck package

Heiko Oberdiek  
<heiko.oberdiek at gmail.com>

2010/03/16 v1.6

## Abstract

The package provides additional rerun warnings if some auxiliary files have changed. It is based on MD5 checksum, provided by pdfTeX.

## Contents

<b>1</b>	<b>Documentation</b>	<b>1</b>
1.1	Options	2
1.2	Interface for class/package authors	2
<b>2</b>	<b>Implementation</b>	<b>3</b>
2.1	Options	3
2.2	Check for checksum feature	4
2.3	Standard <code>..aux</code> files	4
2.4	Rerun check	7
<b>3</b>	<b>Test</b>	<b>8</b>
3.1	Catcode checks for loading	8
<b>4</b>	<b>Installation</b>	<b>10</b>
4.1	Download	10
4.2	Bundle installation	10
4.3	Package installation	10
4.4	Refresh file name databases	11
4.5	Some details for the interested	11
<b>5</b>	<b>History</b>	<b>11</b>
	[2009/12/10 v1.0]	11
	[2009/12/12 v1.1]	11
	[2009/12/18 v1.2]	12
	[2010/01/25 v1.3]	12
	[2010/02/22 v1.4]	12
	[2010/03/15 v1.5]	12
	[2010/03/16 v1.6]	12
<b>6</b>	<b>Index</b>	<b>12</b>

## 1 Documentation

L<sup>A</sup>T<sub>E</sub>X informs the user, when to run L<sup>A</sup>T<sub>E</sub>X again, if the references have changed. It has the old references from the first reading of the `..aux` files already in memory, thus it compares them with the new version of the `..aux` file at the end of the document. However this rerun warnings are not given for the table of contents and other data stored in the `..aux` files or other auxiliary files. Usually many of

these data as the table of contents is not keep in memory. If someone wants to detect changes, he has either to keep the data in memory. This does not scale well with huge documents. Or he copies the file before they are changed. Slow I/O operations cost time.

Since version 1.30.0 pdfTeX provides `\pdfmdfivesum` and `\pdffilesize`. These features are also available in LuaTeX, provided by package `pdftexcmds`. Thus this package `rerunfilecheck` uses these features to detect file changes. This saves the packages from keeping the whole files in memory or in file copies. The drawback are different files with the same size and the same MD5 checksum (seldom, hopefully).

## 1.1 Options

All options are key value options of boolean type. No option or `true` turns an option on, `false` disables an option.

**mainaux:** Check the main `..aux` file.

**partaux:** Check the `..aux` files from `\include` files.

**starttoc:** Add the rerun checks in `\@starttoc` that is called by `\tableofcontents`, `\listoffigures`, ...

**index, glossary:** L<sup>A</sup>T<sub>E</sub>X's original `\makeindex` and `\makeglossary` are redefined to add the rerun checks. The options do not have an effect, if `\makeindex`/`\makeglossary` are already called or if a package or class had redefined or will redefine them.

**aux:** This option turns all previous options on or off. "aux" means auxiliary file.

The default for the options is `false`, because some internals must be redefined to insert the rerun checks. The options can be set in `\usepackage` or the configuration file `rerunfilecheck.cfg`. Global options are ignored (since 1.4).

`\RerunFileCheckSetup {⟨key value list⟩}`

Options can also be set using `\RerunFileCheckSetup`. Currently all options are disabled after the package is loaded. Thus `\RerunFileCheckSetup` makes sense in the configuration file only.

Example for the configuration file:

```

1 (*cfg)
2 \ProvidesFile{rerunfilecheck.cfg}[2010/03/16 Default configuration]%
3 \RerunFileCheckSetup{aux}
4 </cfg>

```

## 1.2 Interface for class/package authors

`\RerunFileCheck {⟨file⟩} {⟨file closing action⟩} {⟨rerun warning⟩}`

If you want to add a rerun check, call `\RerunFileCheck` right before an output file is opened for writing. The macro first remembers the current checksum of `⟨file⟩`. The file is checked again right before the end of the job. Macro `\AtVeryEndDocument` of package `atveryend` is used to place the check after the main aux file is closed in `\end{document}`. Before reading the file again, it must be closed. Provide the code for closing in argument `⟨file closing action⟩`. Do not forget `\immediate` before `\openout`. Otherwise the closing action would be delayed to the next shipout that never happens (the last page is already shipped out). If the file has changed, `\RerunFileCheck` informs the user with a warning that the file has changed and says the magic word "Rerun". If the last argument `⟨rerun warning⟩` is not empty, then the rerun sentence is replaced by it. Usually

the phrase “to get something right” is added. As example the relevant part of the redefined `\makeindex` is shown, see package code:

```

\newwrite\@indexfile
\RunFileCheck{\jobname.idx}{%
  \immediate\closeout\@indexfile
}%
  Rerun LaTeX/makeindex to get index right%
}%
\immediate\openout\@indexfile=\jobname.idx %

```

## 2 Implementation

```

5 (*package)
6 \begingroup
7 \catcode123 1 % {
8 \catcode125 2 % }
9 \def\x{\endgroup
10 \expandafter\edef\csname ReFiCh@AtEnd\endcsname{%
11 \catcode35 \the\catcode35\relax
12 \catcode64 \the\catcode64\relax
13 \catcode123 \the\catcode123\relax
14 \catcode125 \the\catcode125\relax
15 }%
16 }%
17 \x
18 \catcode35 6 % #
19 \catcode64 11 % @
20 \catcode123 1 % {
21 \catcode125 2 % }
22 \def\TMP@EnsureCode#1#2{%
23 \edef\ReFiCh@AtEnd{%
24 \ReFiCh@AtEnd
25 \catcode#1 \the\catcode#1\relax
26 }%
27 \catcode#1 #2\relax
28 }
29 \TMP@EnsureCode{39}{12}% '
30 \TMP@EnsureCode{40}{12}% (
31 \TMP@EnsureCode{41}{12}% )
32 \TMP@EnsureCode{42}{12}% *
33 \TMP@EnsureCode{44}{12}% ,
34 \TMP@EnsureCode{46}{12}% .
35 \TMP@EnsureCode{47}{12}% /
36 \TMP@EnsureCode{58}{12}% :
37 \TMP@EnsureCode{59}{12}% ;
38 \TMP@EnsureCode{60}{12}% <
39 \TMP@EnsureCode{61}{12}% =
40 \TMP@EnsureCode{62}{12}% >
41 \TMP@EnsureCode{91}{12}% [
42 \TMP@EnsureCode{93}{12}% ]
43 \TMP@EnsureCode{96}{12}% '
44 \g@addto@macro\ReFiCh@AtEnd{\endinput}

Package identification.
45 \NeedsTeXFormat{LaTeX2e}
46 \ProvidesPackage{rerunfilecheck}%
47 [2010/03/16 v1.6 Rerun checks for auxiliary files (HO)]

```

### 2.1 Options

```

48 \RequirePackage{kvoptions}[2010/02/22]
49 \SetupKeyvalOptions{%

```

```

50 family=rerunfilecheck,%
51 prefix=ReFiCh%
52 }

\RerunFileCheckSetup

53 \newcommand*\RerunFileCheckSetup{%
54   \setkeys{rerunfilecheck}%
55 }

56 \DeclareBoolOption{mainaux}
57 \DeclareBoolOption{partaux}
58 \DeclareBoolOption{starttoc}
59 \DeclareBoolOption{index}
60 \DeclareBoolOption{glossary}
61 \define@key{rerunfilecheck}{aux}[true]{%
62   \RerunFileCheckSetup{%
63     mainaux={#1},%
64     partaux={#1},%
65     starttoc={#1},%
66     index={#1},%
67     glossary={#1}%
68   }%
69 }

70 \InputIfFileExists{rerunfilecheck.cfg}{-}{-}
71 \ProcessLocalKeyvalOptions*

```

\ReFiCh@DisableOption

```

72 \def\ReFiCh@DisableOption{%
73   \DisableKeyvalOption[%
74     action=warning,%
75     package=rerunfilecheck%
76   ]{rerunfilecheck}%
77 }

```

## 2.2 Check for checksum feature

```

78 \RequirePackage{infwarerr}[2007/09/09]
79 \RequirePackage{pdftexcmds}[2009/04/10]

80 \begingroup\expandafter\expandafter\expandafter\endgroup
81 \expandafter\ifx\csname pdf@filemdfivesum\endcsname\relax
82   \@PackageInfoNoLine{rerunfilecheck}{%
83     Feature \string\pdfmdfivesum\space is not available\MessageBreak
84     (e.g. pdfTeX or LuaTeX with package 'pdftexcmds')\MessageBreak
85     Therefore file contents cannot be checked efficiently\MessageBreak
86     and the loading of the package is aborted%
87   }%
88   \newcommand*\RerunFileCheck}[3]{}%
89   \renewcommand*\RerunFileCheckSetup}[1]{}%
90   \expandafter\ReFiCh@AtEnd
91 \fi

```

## 2.3 Standard ..aux files

```

92 \ifReFiCh@partaux
93   \let\ReFiCh@org@include\@include
94   \def\@include#1 {%
95     \if@filesw
96       \RerunFileCheck{#1.aux}{-}{-}%
97     \fi
98     \ReFiCh@org@include{#1} %
99   }%
100 \fi

```

```

101 \ifReFiCh@mainaux
102 \AtBeginDocument{%
103   \ReFiCh@mainauxfalse
104 }%
105 \ifReFiCh@mainaux
106 \AtEndOfPackage{%
107   \RerunFileCheck{\jobname.aux}{-}{-}%
108 }%
109 \else
110 \if@filesw
111   \@PackageWarningNoLine{rerunfilecheck}{%
112     Main aux file check is disabled,\MessageBreak
113     because the file is already opened.\MessageBreak
114     Load the package before \string\begin{document}%
115   }%
116 \fi
117 \fi
118 \fi
119 \ifReFiCh@starttoc
120 \let\ReFiCh@org@starttoc\@starttoc
121 \def\@starttoc#1{%
122   \if@filesw
123     \RerunFileCheck{\jobname.#1}{-}{-}%
124     \@ifundefined{tf@#1}{%
125       }{%
126         \immediate\closeout\csname tf@#1\endcsname
127       }%
128     }{}%
129   \fi
130   \ReFiCh@org@starttoc{#1}%
131 }%
132 \fi
133 \ifReFiCh@index
134 \ifx\makeindex\@empty
135   \@PackageWarningNoLine{rerunfilecheck}{%
136     Option 'index' ignored,\MessageBreak
137     because \string\makeindex\space has already been called%
138   }%
139 \else
140 \def\ReFiCh@temp{%
141   \newwrite\@indexfile
142   \immediate\openout\@indexfile=\jobname.idx %
143   \def\index{%
144     \@bsphack
145     \begingroup
146     \@sanitize
147     \@wrindex
148   }%
149   \typeout{Writing index file \jobname.idx}%
150   \let\makeindex\@empty
151 }%
152 \ifx\ReFiCh@temp\makeindex
153 \def\makeindex{%
154   \newwrite\@indexfile
155   \RerunFileCheck{\jobname.idx}{-}{-}%
156   \immediate\closeout\@indexfile
157 }{%
158   Rerun LaTeX/makeindex to get index right%
159 }%
160 \immediate\openout\@indexfile=\jobname.idx %
161 \def\index{%
162   \@bsphack

```

```

163     \begingroup
164     \@sanitize
165     \@wrindex
166     }%
167     \typeout{Writing index file \jobname.idx}%
168     \let\makeindex\@empty
169     }%
170     \else
171     \@PackageInfoNoLine{rerunfilecheck}{%
172     Option 'index': unsupported version of \string\makeindex
173     }%
174     \fi
175     \fi
176 \fi
177 \ifReFiCh@glossary
178 \ifx\makeglossary\@empty
179     \@PackageWarningNoLine{rerunfilecheck}{%
180     Option 'glossary' ignored,\MessageBreak
181     because \string\makeglossary\space has already been called%
182     }%
183     \else
184     \def\ReFiCh@temp{%
185     \newwrite\@glossaryfile
186     \immediate\openout\@glossaryfile=\jobname.glo %
187     \def\glossary{%
188     \@bsphack
189     \begingroup
190     \@sanitize
191     \@wrglossary
192     }%
193     \typeout{Writing glossary file \jobname.glo }%
194     \let\makeglossary\@empty
195     }%
196     \ifx\ReFiCh@temp\makeglossary
197     \def\ReFiCh@temp{%
198     \newwrite\@glossaryfile
199     \RerunFileCheck{\jobname.glo}{%
200     \immediate\closeout\@glossaryfile
201     }{%
202     Rerun LaTeX/makeindex to get glossary right%
203     }%
204     \immediate\openout\@glossaryfile=\jobname.glo %
205     \def\glossary{%
206     \@bsphack
207     \begingroup
208     \@sanitize
209     \@wrglossary
210     }%
211     \typeout{Writing glossary file \jobname.glo}%
212     \let\makeglossary\@empty
213     }%
214     \else
215     \@PackageInfoNoLine{rerunfilecheck}{%
216     Option 'glossary': unsupported version of \string\makeglossary
217     }%
218     \fi
219     \fi
220 \fi
221 \ReFiCh@DisableOption{mainaux}
222 \ReFiCh@DisableOption{partaux}
223 \ReFiCh@DisableOption{starttoc}
224 \ReFiCh@DisableOption{index}

```

```
225 \ReFiCh@DisableOption{glossary}
226 \ReFiCh@DisableOption{aux}
```

## 2.4 Rerun check

```
227 \RequirePackage{atveryend}[2009/12/07]
228 \RequirePackage{uniquecounter}[2009/12/18]
```

\ReFiCh@Checksum

```
229 \begingroup\expandafter\expandafter\expandafter\endgroup
230 \expandafter\ifx\csname pdf@filesize\endcsname\relax
231   \def\ReFiCh@Checksum{%
232     \pdf@filemdfivesum
233   }%
234 \else
235   \def\ReFiCh@Checksum#1{%
236     \pdf@filemdfivesum{#1}%
237     \ReFiCh@Separator
238     \pdf@filesize{#1}%
239   }%
240 \fi
```

\ReFiCh@NoFile

```
241 \def\ReFiCh@Separator{;}
```

\ReFiCh@NoFile

```
242 \def\ReFiCh@NoFile{<no file>}
243 \UniqueCounterNew{rerunfilecheck}
```

\RerunFileCheck

```
244 \newcommand*{\RerunFileCheck}{%
245   \UniqueCounterCall{rerunfilecheck}\ReFiCh@RerunFileCheck
246 }
```

\ReFiCh@RerunFileCheck

```
247 \def\ReFiCh@RerunFileCheck#1{%
248   \expandafter\ReFiCh@@RerunFileCheck\csname ReFiCh@#1\endcsname
249 }
```

\ReFiCh@Check

```
250 \def\ReFiCh@Check#1#2#3{%
251 %   \IfFileExists{#3}{%
252   #1\edef#2{\ReFiCh@Checksum{#3}}%
253   \ifx#2\ReFiCh@Separator
254     #1\let#2\ReFiCh@NoFile
255   \fi
256 % }{%
257 %   #1\let#2\ReFiCh@NoFile
258 % }%
259 }
```

\ReFiCh@@RerunFileCheck

```
260 \def\ReFiCh@@RerunFileCheck#1#2#3#4{%
261   \ReFiCh@Check\global#1{#2}%
262   \AtVeryEndDocument{%
263     \begingroup
264     #3%
265     \ReFiCh@Check}\x{#2}%
266     \ifx#1\x
267       \@PackageInfoNoLine{rerunfilecheck}{%
268         File ‘#2’ has not changed.\MessageBreak
```

```

269     Checksum: \x
270   }%
271   \else
272     \ifnum
273       \ReFiCh@IsAux#2\relax.aux\relax\@nil
274       \ifx#1\ReFiCh@NoFile 1\else 0\fi
275       \ifx\x\ReFiCh@AuxEmptyUnix 1%
276       \else
277         \ifx\x\ReFiCh@AuxEmptyDos 1\fi
278       \fi
279     =111 %
280     \@PackageInfoNoLine{rerunfilecheck}{%
281       File ‘#2’ is empty .aux file.\MessageBreak
282       Before: #1\MessageBreak
283       After: \space\x
284     }%
285   \else
286     \@PackageWarningNoLine{rerunfilecheck}{%
287       File ‘#2’ has changed.%
288       \ifx\#4\%
289         \space Rerun%
290       \else
291         \MessageBreak
292         #4%
293       \fi
294     }%
295     \@PackageInfoNoLine{rerunfilecheck}{%
296       Checksums for ‘#2’:\MessageBreak
297       Before: #1\MessageBreak
298       After: \space\x
299     }%
300   \fi
301 \fi
302 \endgroup
303 }%
304 }

305 \def\ReFiCh@IsAux#1.aux\relax#2\@nil{%
306   \ifx\hbox#2\hbox
307     0%
308   \else
309     1%
310   \fi
311 }

312 \def\ReFiCh@AuxEmptyUnix{A94A2480D3289E625EEA47CD1B285758;8}%
313 \@onelevel@sanitize\ReFiCh@AuxEmptyUnix

314 \def\ReFiCh@AuxEmptyDos{A62A15ECE803E2EBE94952FCC9933BC0;9}%
315 \@onelevel@sanitize\ReFiCh@AuxEmptyDos

316 \ReFiCh@AtEnd
317 \endpackage

```

### 3 Test

```
i*test1_i i/test1_i
```

#### 3.1 Catcode checks for loading

```

318 (*test1)
319 \catcode‘\{=1 %
320 \catcode‘\}=2 %

```



```

321 \catcode'\#=6 %
322 \catcode'\@=11 %
323 \expandafter\ifx\csname count@\endcsname\relax
324   \countdef\count@=255 %
325 \fi
326 \expandafter\ifx\csname @gobble\endcsname\relax
327   \long\def@gobble#1{}%
328 \fi
329 \expandafter\ifx\csname @firstofone\endcsname\relax
330   \long\def\@firstofone#1{#1}%
331 \fi
332 \expandafter\ifx\csname loop\endcsname\relax
333   \expandafter\@firstofone
334 \else
335   \expandafter@gobble
336 \fi
337 {%
338   \def\loop#1\repeat{%
339     \def\body{#1}%
340     \iterate
341   }%
342   \def\iterate{%
343     \body
344     \let\next\iterate
345   \else
346     \let\next\relax
347   \fi
348   \next
349 }%
350 \let\repeat=\fi
351 }%
352 \def\RestoreCatcodes{}
353 \count@=0 %
354 \loop
355   \edef\RestoreCatcodes{%
356     \RestoreCatcodes
357     \catcode\the\count@=\the\catcode\count@\relax
358   }%
359 \ifnum\count@<255 %
360   \advance\count@ 1 %
361 \repeat
362
363 \def\RangeCatcodeInvalid#1#2{%
364   \count@=#1\relax
365   \loop
366     \catcode\count@=15 %
367   \ifnum\count@<#2\relax
368     \advance\count@ 1 %
369   \repeat
370 }
371 \expandafter\ifx\csname LoadCommand\endcsname\relax
372   \def\LoadCommand{\input rerunfilecheck.sty\relax}%
373 \fi
374 \def\Test{%
375   \RangeCatcodeInvalid{0}{47}%
376   \RangeCatcodeInvalid{58}{64}%
377   \RangeCatcodeInvalid{91}{96}%
378   \RangeCatcodeInvalid{123}{255}%
379   \catcode'\@=12 %
380   \catcode'\=0 %
381   \catcode'\{=1 %
382   \catcode'\}=2 %

```

```

383 \catcode'\#=6 %
384 \catcode'\ [=12 %
385 \catcode'\]=12 %
386 \catcode'\%=14 %
387 \catcode'\ =10 %
388 \catcode13=5 %
389 \LoadCommand
390 \RestoreCatcodes
391 }
392 \Test
393 \csname @@end\endcsname
394 \end
395 </test1>

```

## 4 Installation

### 4.1 Download

**Package.** This package is available on CTAN<sup>1</sup>:

[CTAN:macros/latex/contrib/oberdiek/rerunfilecheck.dtx](#) The source file.

[CTAN:macros/latex/contrib/oberdiek/rerunfilecheck.pdf](#) Documentation.

**Bundle.** All the packages of the bundle ‘oberdiek’ are also available in a TDS compliant ZIP archive. There the packages are already unpacked and the documentation files are generated. The files and directories obey the TDS standard.

[CTAN:install/macros/latex/contrib/oberdiek.tds.zip](#)

*TDS* refers to the standard “A Directory Structure for  $\TeX$  Files” ([CTAN:tds/tds.pdf](#)). Directories with `texmf` in their name are usually organized this way.

### 4.2 Bundle installation

**Unpacking.** Unpack the `oberdiek.tds.zip` in the TDS tree (also known as `texmf` tree) of your choice. Example (linux):

```
unzip oberdiek.tds.zip -d ~/texmf
```

**Script installation.** Check the directory `TDS:scripts/oberdiek/` for scripts that need further installation steps. Package `attachfile2` comes with the Perl script `pdfatfi.pl` that should be installed in such a way that it can be called as `pdfatfi`. Example (linux):

```
chmod +x scripts/oberdiek/pdfatfi.pl
cp scripts/oberdiek/pdfatfi.pl /usr/local/bin/
```

### 4.3 Package installation

**Unpacking.** The `.dtx` file is a self-extracting docstrip archive. The files are extracted by running the `.dtx` through plain  $\TeX$ :

```
tex rerunfilecheck.dtx
```

---

<sup>1</sup><ftp://ftp.ctan.org/tex-archive/>

**TDS.** Now the different files must be moved into the different directories in your installation TDS tree (also known as `texmf` tree):

```
rerunfilecheck.sty      → tex/latex/oberdiek/rerunfilecheck.sty
rerunfilecheck.pdf      → doc/latex/oberdiek/rerunfilecheck.pdf
rerunfilecheck-example.cfg → doc/latex/oberdiek/rerunfilecheck-example.cfg
test/rerunfilecheck-test1.tex → doc/latex/oberdiek/test/rerunfilecheck-test1.tex
rerunfilecheck.dtx      → source/latex/oberdiek/rerunfilecheck.dtx
```

If you have a `docstrip.cfg` that configures and enables `docstrip`'s TDS installing feature, then some files can already be in the right place, see the documentation of `docstrip`.

## 4.4 Refresh file name databases

If your  $\TeX$  distribution (te $\TeX$ , mi $\TeX$ , ...) relies on file name databases, you must refresh these. For example, te $\TeX$  users run `texhash` or `mktexlsr`.

## 4.5 Some details for the interested

**Attached source.** The PDF documentation on CTAN also includes the `.dtx` source file. It can be extracted by AcrobatReader 6 or higher. Another option is `pdftk`, e.g. unpack the file into the current directory:

```
pdftk rerunfilecheck.pdf unpack_files output .
```

**Unpacking with  $\LaTeX$ .** The `.dtx` chooses its action depending on the format:

**plain  $\TeX$ :** Run `docstrip` and extract the files.

**$\LaTeX$ :** Generate the documentation.

If you insist on using  $\LaTeX$  for `docstrip` (really, `docstrip` does not need  $\LaTeX$ ), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{rerunfilecheck.dtx}
```

Do not forget to quote the argument according to the demands of your shell.

**Generating the documentation.** You can use both the `.dtx` or the `.drv` to generate the documentation. The process can be configured by the configuration file `ltxdoc.cfg`. For instance, put this line into this file, if you want to have A4 as paper format:

```
\PassOptionsToClass{a4paper}{article}
```

An example follows how to generate the documentation with `pdf $\LaTeX$` :

```
pdflatex rerunfilecheck.dtx
makeindex -s gind.ist rerunfilecheck.idx
pdflatex rerunfilecheck.dtx
makeindex -s gind.ist rerunfilecheck.idx
pdflatex rerunfilecheck.dtx
```

## 5 History

[2009/12/10 v1.0]

- The first version.

[2009/12/12 v1.1]

- Short info shortened.

[2009/12/18 v1.2]

- Required date for package `uniquecounter` updated because of bug in this package.

[2010/01/25 v1.3]

- Moved from `TDS:*/generic/*` to `TDS:*/latex/*`.

[2010/02/22 v1.4]

- The options of this package are recognized only if they are package options. Global options are ignored. This avoids name clashes with class and other package options (for example, class option `'index=totoc'`).

[2010/03/15 v1.5]

- Call of `\pdfvivesum` is wrapped in `\IfFileExists` to avoid calls of `mktexTeX` if this feature is enabled. However `\IfFileExists` has file name limitations.

[2010/03/16 v1.6]

- Reverted to version 1.4 and `\IfFileExists` wrapper of version 1.5 is removed.

## 6 Index

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

Symbols	
<code>\#</code> .....	<a href="#">321, 383</a>
<code>\%</code> .....	<a href="#">386</a>
<code>\@</code> .....	<a href="#">322, 379</a>
<code>\@PackageInfoNoLine</code> .....	
.....	<a href="#">82, 171, 215, 267, 280, 295</a>
<code>\@PackageWarningNoLine</code> .....	
.....	<a href="#">111, 135, 179, 286</a>
<code>\@bsphack</code> .....	<a href="#">144, 162, 188, 206</a>
<code>\@empty</code> ...	<a href="#">134, 150, 168, 178, 194, 212</a>
<code>\@firstofone</code> .....	<a href="#">330, 333</a>
<code>\@glossaryfile</code> .....	<a href="#">185, 186, 198, 200, 204</a>
<code>\@gobble</code> .....	<a href="#">327, 335</a>
<code>\@ifundefined</code> .....	<a href="#">124</a>
<code>\@include</code> .....	<a href="#">93, 94</a>
<code>\@indexfile</code> ...	<a href="#">141, 142, 154, 156, 160</a>
<code>\@nil</code> .....	<a href="#">273, 305</a>
<code>\@onelevel@sanitize</code> .....	<a href="#">313, 315</a>
<code>\@sanitize</code> .....	<a href="#">146, 164, 190, 208</a>
<code>\@starttoc</code> .....	<a href="#">120, 121</a>
<code>\@wrglossary</code> .....	<a href="#">191, 209</a>
<code>\@wrindex</code> .....	<a href="#">147, 165</a>
<code>\[</code> .....	<a href="#">384</a>
<code>\]</code> .....	<a href="#">288, 380</a>
<code>\{</code> .....	<a href="#">319, 381</a>
<code>\}</code> .....	<a href="#">320, 382</a>
<code>\]</code> .....	<a href="#">385</a>
<code>\_</code> .....	<a href="#">387</a>
<b>A</b>	
<code>\advance</code> .....	<a href="#">360, 368</a>
<code>\AtBeginDocument</code> .....	<a href="#">102</a>
<code>\AtEndOfPackage</code> .....	<a href="#">106</a>
<code>\AtVeryEndDocument</code> .....	<a href="#">262</a>
<b>B</b>	
<code>\begin</code> .....	<a href="#">114</a>
<code>\body</code> .....	<a href="#">339, 343</a>
<b>C</b>	
<code>\catcode</code> ...	<a href="#">7, 8, 11, 12, 13, 14, 18,</a> <a href="#">19, 20, 21, 25, 27, 319, 320, 321,</a> <a href="#">322, 357, 366, 379, 380, 381,</a> <a href="#">382, 383, 384, 385, 386, 387, 388</a>
<code>\closeout</code> .....	<a href="#">126, 156, 200</a>
<code>\count@</code> .....	<a href="#">324, 353,</a> <a href="#">357, 359, 360, 364, 366, 367, 368</a>
<code>\countdef</code> .....	<a href="#">324</a>
<code>\csname</code> .....	<a href="#">10, 81, 126, 230,</a> <a href="#">248, 323, 326, 329, 332, 371, 393</a>
<b>D</b>	
<code>\DeclareBoolOption</code> ..	<a href="#">56, 57, 58, 59, 60</a>
<code>\define@key</code> .....	<a href="#">61</a>
<code>\DisableKeyvalOption</code> .....	<a href="#">73</a>

<b>E</b>		<b>P</b>	
<code>\end</code> .....	394	<code>\pdf@filemdfivesum</code> .....	232, 236
<code>\endcsname</code> .....	10, 81, 126, 230, 248, 323, 326, 329, 332, 371, 393	<code>\pdf@filesize</code> .....	238
<code>\endinginput</code> .....	44	<code>\pdfmdfivesum</code> .....	83
<b>G</b>		<code>\ProcessLocalKeyvalOptions</code> .....	71
<code>\g@addto@macro</code> .....	44	<code>\ProvidesFile</code> .....	2
<code>\glossary</code> .....	187, 205	<code>\ProvidesPackage</code> .....	46
<b>H</b>		<b>R</b>	
<code>\hbox</code> .....	306	<code>\RangeCatcodeInvalid</code> .....	363, 375, 376, 377, 378
<b>I</b>		<code>\ReFiCh@@RerunFileCheck</code> ...	248, 260
<code>\if@filesw</code> .....	95, 110, 122	<code>\ReFiCh@AtEnd</code> .....	23, 24, 44, 90, 316
<code>\IfFileExists</code> .....	251	<code>\ReFiCh@AuxEmptyDos</code> ...	277, 314, 315
<code>\ifnum</code> .....	272, 359, 367	<code>\ReFiCh@AuxEmptyUnix</code> ..	275, 312, 313
<code>\ifReFiCh@glossary</code> .....	177	<code>\ReFiCh@Check</code> .....	250, 261, 265
<code>\ifReFiCh@index</code> .....	133	<code>\ReFiCh@CheckSum</code> .....	229, 252
<code>\ifReFiCh@mainaux</code> .....	101, 105	<code>\ReFiCh@DisableOption</code> .....	72, 221, 222, 223, 224, 225, 226
<code>\ifReFiCh@partaux</code> .....	92	<code>\ReFiCh@IsAux</code> .....	273, 305
<code>\ifReFiCh@starttoc</code> .....	119	<code>\ReFiCh@mainauxfalse</code> .....	103
<code>\ifx</code> .....	81, 134, 152, 178, 196, 230, 253, 266, 274, 275, 277, 288, 306, 323, 326, 329, 332, 371	<code>\ReFiCh@NoFile</code> 241, 242, 254, 257, 274	
<code>\immediate</code> .....	126, 142, 156, 160, 186, 200, 204	<code>\ReFiCh@org@include</code> .....	93, 98
<code>\index</code> .....	143, 161	<code>\ReFiCh@org@starttoc</code> .....	120, 130
<code>\input</code> .....	372	<code>\ReFiCh@RerunFileCheck</code> ...	245, 247
<code>\InputIfFileExists</code> .....	70	<code>\ReFiCh@Separator</code> .....	237, 241, 253
<code>\iterate</code> .....	340, 342, 344	<code>\ReFiCh@temp</code> ..	140, 152, 184, 196, 197
<b>J</b>		<code>\renewcommand</code> .....	89
<code>\jobname</code> ..	107, 123, 142, 149, 155, 160, 167, 186, 193, 199, 204, 211	<code>\repeat</code> .....	338, 350, 361, 369
<b>L</b>		<code>\RequirePackage</code> ..	48, 78, 79, 227, 228
<code>\LoadCommand</code> .....	372, 389	<code>\RerunFileCheck</code> .....	2, 88, 96, 107, 123, 155, 199, 244
<code>\loop</code> .....	338, 354, 365	<code>\RerunFileCheckSetup</code> ..	2, 3, 53, 62, 89
<b>M</b>		<code>\RestoreCatcodes</code> ..	352, 355, 356, 390
<code>\makeglossary</code> .....	178, 181, 194, 196, 212, 216	<b>S</b>	
<code>\makeindex</code> .....	134, 137, 150, 152, 153, 168, 172	<code>\setkeys</code> .....	54
<code>\MessageBreak</code> .....	83, 84, 85, 112, 113, 136, 180, 268, 281, 282, 291, 296, 297	<code>\SetupKeyvalOptions</code> .....	49
<b>N</b>		<code>\space</code> .....	83, 137, 181, 283, 289, 298
<code>\NeedsTeXFormat</code> .....	45	<b>T</b>	
<code>\newcommand</code> .....	53, 88, 244	<code>\Test</code> .....	374, 392
<code>\newwrite</code> .....	141, 154, 185, 198	<code>\the</code> .....	11, 12, 13, 14, 25, 357
<code>\next</code> .....	344, 346, 348	<code>\TMP@EnsureCode</code> 22, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43	
<b>O</b>		<code>\typeout</code> .....	149, 167, 193, 211
<code>\openout</code> .....	142, 160, 186, 204	<b>U</b>	
<b>X</b>		<code>\UniqueCounterCall</code> .....	245
<code>\x</code> 9, 17, 265, 266, 269, 275, 277, 283, 298		<code>\UniqueCounterNew</code> .....	243