

The `kvsetkeys` package

Heiko Oberdiek
<oberdiek@uni-freiburg.de>

2006/10/19 v1.1

Abstract

Package `kvsetkeys` provides `\kvsetkeys`, a variant of package `keyval`'s `\setkeys`. It allows to specify a handler that deals with unknown options. Active commas and equal signs may be used (e.g. see `babel`'s shorthands) and only one level of curly braces is removed from the values.

Contents

1	Documentation	1
1.1	Normalizing key value lists	2
1.2	Parsing key value lists	2
1.3	Processing key value pairs	3
1.4	Default family handler	3
1.5	Do it all	3
2	Example	3
3	Implementation	4
3.1	Identification	4
3.2	Normalizing key value lists	6
3.3	Parsing key value lists	8
3.4	Processing key value pairs	9
3.5	Error handling	10
3.6	Do it all	10
4	Installation	11
4.1	Download	11
4.2	Bundle installation	11
4.3	Package installation	11
4.4	Refresh file name databases	11
4.5	Some details for the interested	11
5	References	12
6	History	12
	[2006/03/06 v1.0]	12
	[2006/10/19 v1.1]	12
7	Index	12

1 Documentation

`\kvsetkeys` can be used as replacement for `keyval`'s `\setkeys`. Also it uses the same syntax. Before I describe `\kvsetkeys` in more detail, first I want to explain, how this package deals with key value lists. For the package also provides low level interfaces that can be used by package authors.

1.1 Normalizing key value lists

`\kv@normalize {⟨key value list⟩}`

Specifying key value lists, the user usually wants to have nice formatted source code, e.g.:

```
\hypersetup{
  pdftitle   = {...},
  pdfsubject = {...},
  pdfauthor  = {...},
  pdfkeywords = {...},
  ...
}
```

Thus there can be spaces around keys, around = or around the value. Also empty entries are possible by too many commas. Therefore these spaces and empty entries are silently removed by package `keyval` and this package. Whereas the contents of the value can be protected by curly braces, especially if spaces or commas are used inside, a key name must not use spaces or other syntax characters.

`\kv@normalize` takes a key value list and performs the cleanup:

- Spaces are removed.
- Syntax characters (comma and equal sign) that are active are replaced by the same characters with standard catcode. (Example: `babel`'s language option `turkish` uses the equal sign as active shorthand character.)

The result is stored in `\kv@list`, e.g.:

```
\kv@list → ,pdftitle={},pdfsubject={},...,
```

Empty entries are removed later. In fact it adds a comma at the begin and end to protect the last value and an easier implementation.

1.2 Parsing key value lists

`\kv@parse {⟨key value list⟩} {⟨processor⟩}`

It is easier to parse a normalized list, thus `\kv@parse` normalizes the list and calls `\kv@parse@normalized`.

`\kv@parse@normalized {⟨key value list⟩} {⟨processor⟩}`

Now the key value list is split into single key value pairs. For further processing the key and value are given as arguments for the `⟨processor⟩`:

```
⟨processor⟩ {⟨key⟩} {⟨value⟩}
```

Also key and value are stored in macro names:

- `\kv@key` stores the key.
- `\kv@value` stores the value or if the value was not specified it has the meaning `\relax`.

The behaviour in pseudo code:

```
foreach (⟨key⟩, ⟨value⟩) in (⟨key value list⟩)
  \kv@key := ⟨key⟩
  \kv@value := ⟨value⟩
  ⟨processor⟩ {⟨key⟩} {⟨value⟩}
```

1.3 Processing key value pairs

`\kv@processor@default {⟨family⟩} {⟨key⟩} {⟨value⟩}`

There are many possibilities to process key value pairs. `\kv@processor@default` is the processor used in `\kvsetkeys`. It reimplements and extends the behaviour of `keyval`'s `\setkeys`. In case of unknown keys `\setkeys` raise an error. This processor, however, calls a handler instead, if it is provided by the family.

The behaviour in pseudo code:

```
if ⟨key⟩ exists
  call the keyval code of ⟨key⟩
else
  if ⟨handler⟩ for ⟨family⟩ exists
    ⟨handler⟩ {⟨key⟩} {⟨value⟩}
  else
    raise unknown key error
fi
fi
```

1.4 Default family handler

`\kv@processor@default` calls `⟨handler⟩`, the default handler for the family, if the key does not exist in the family. The handler is called with two arguments, the key and the value. It can be defined with `\kv@set@family@handler`:

`\kv@set@family@handler {⟨family⟩} {⟨handler definition⟩}`

This sets the default family handler for the keyval family `⟨family⟩`. Inside `⟨handler definition⟩` #1 stands for the key and #2 is the value. Also `\kv@key` and `\kv@value` can be used for the key and the value. If the value is not given, `\kv@value` has the meaning `\relax`.

1.5 Do it all

`\kvsetkeys {⟨family⟩} {⟨key value list⟩}`

The work is done by the previous commands. `\kvsetkeys` just calls them:

```
\kv@parse {⟨key value list⟩} {\kv@processor@default {⟨family⟩}}
```

Thus you can replace `\setkeys` of package `keyval` by the key value parser of this package:

```
\renewcommand*{\setkeys}{\kvsetkeys}
or
\let\setkeys\kvsetkeys
```

2 Example

The following example prints a short piece of HTML code using the tabbing environment for indenting purpose and a key value syntax for specifying the attributes of an HTML tag. The example illustrates the use of a default family handler.

```
1 ⟨*example⟩
2 \documentclass{article}
3 \usepackage[T1]{fontenc}
4 \usepackage{kvsetkeys}
```

```

5 \usepackage{keyval}
6
7 \makeatletter
8 \newcommand*{\tag}[2] [] {%
9   % #1: attributes
10  % #2: tag name
11  \begin{group}
12    \toks@={}%
13    \let\endslash\@empty
14    \kvsetkeys{tag}{#1}%
15    \texttt{%
16      \textless #2\the\toks@\@endslash\textgreater
17    }%
18  \end{group}
19 }
20 \kv@set@family@handler{tag}{%
21   % #1: key
22   % #2: value
23   \toks@\expandafter{%
24     \the\toks@
25     \space
26     #1=\string"#2\string"%
27   }%
28 }
29 \define@key{tag}{/}[] {%
30   \def\@endslash{/}%
31 }
32 \makeatother
33
34 \begin{document}
35 \begin{tabbing}
36   \mbox{}}\qqquad=\qqquad=\kill
37   \tag{html}\\
38   \>\dots\\
39   \>\tag[border=1]{table}\\
40   \>\>\tag[width=200, span=3, /]{colgroup}\\
41   \>\>\dots\\
42   \>\tag{/table}\\
43   \>\dots\\
44   \tag{/html}\\
45 \end{tabbing}
46 \end{document}
47 \end{example}

```

3 Implementation

3.1 Identification

48 **⌈package⌋**

Reload check, especially if the package is not used with L^AT_EX.

```

49 \begingroup
50 \catcode44 12 % ,
51 \catcode45 12 % -
52 \catcode46 12 % .
53 \catcode58 12 % :
54 \catcode64 11 % @
55 \expandafter\let\expandafter\x\csname ver@kvsetkeys.sty\endcsname
56 \ifcase 0%
57 \ifx\x\relax % plain
58 \else
59 \ifx\x\empty % LaTeX

```

```

60     \else
61     1%
62     \fi
63     \fi
64 \else
65 \expandafter\ifx\csname PackageInfo\endcsname\relax
66 \def\x#1#2{%
67 \immediate\write-1{Package #1 Info: #2.}%
68 }%
69 \else
70 \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
71 \fi
72 \x{kvsetkeys}{The package is already loaded}%
73 \endgroup
74 \expandafter\endinput
75 \fi
76 \endgroup
Package identification:
77 \begingroup
78 \catcode44 12 % ,
79 \catcode45 12 % -
80 \catcode46 12 % .
81 \catcode58 12 % :
82 \catcode64 11 % @
83 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
84 \def\x#1#2#3[#4]{\endgroup
85 \immediate\write-1{Package: #3 #4}%
86 \xdef#1{#4}%
87 }%
88 \else
89 \def\x#1#2[#3]{\endgroup
90 #2[#{#3}]%
91 \ifx#1\relax
92 \xdef#1{#3}%
93 \fi
94 }%
95 \fi
96 \expandafter\x\csname ver@kvsetkeys.sty\endcsname
97 \ProvidesPackage{kvsetkeys}%
98 [2006/10/19 v1.1 Key value parser with default handler support (H0)]
99 \expandafter\edef\csname KVS@endinput\endcsname{%
100 \catcode39 \the\catcode39 % '
101 \catcode44 \the\catcode44 % ,
102 \catcode61 \the\catcode61 % =
103 \catcode64 \the\catcode64 % @
104 \catcode94 \the\catcode94 % ^
105 \catcode96 \the\catcode96 % '
106 \catcode126 \the\catcode126 % ~
107 \relax
108 \noexpand\endinput
109 }
110 \catcode39 12 % '
111 \catcode44 12 % ,
112 \catcode61 12 % =
113 \catcode64 11 % @
114 \catcode94 7 % ^
115 \catcode96 12 % '
116 \catcode126 13 % ~
117 \def\KVS@empty{}
118 \long\def\@ReturnAfterFi#1\fi{\fi#1}

```

3.2 Normalizing key value lists

`\kv@normalize`

```

119 \def\kv@normalize#1{%
120   \begingroup
121     \toks@{, #1}%
122     \KVS@comma
123     \KVS@equal
124     \KVS@spaceA
125     \KVS@spaceB{ }%
126     \KVS@spaceC
127     \KVS@spaceD{ }%
128     \xdef\kv@global{\the\toks@}%
129   \endgroup
130   \let\kv@list\kv@global
131 }

```

`\KVS@comma` Converts active commas into comma with catcode other. Also adds a comma at the end to protect the last value for next cleanup steps.

```

132 \begingroup
133   \lccode'\,='\",%
134   \lccode'\~='\",%
135 \lowercase{\endgroup
136   \def\KVS@comma{%
137     \toks@\expandafter{\expandafter}\expandafter
138     \KVS@@comma\the\toks@\~\KVS@nil
139   }%
140   \def\KVS@@comma#1~#2\KVS@nil{%
141     \toks@\expandafter{\the\toks@#1,}%
142     \toks2{#2}%
143     \edef\x{\the\toks2}%
144     \ifx\x\KVS@empty
145     \else
146       \@ReturnAfterFi{%
147         \KVS@@comma#2\KVS@nil
148       }%
149     \fi
150   }%
151 }

```

`\KVS@equal` Converts active equal signs into catcode other characters.

```

152 \begingroup
153   \lccode'\,='\",%
154   \lccode'\~='\",%
155 \lowercase{\endgroup
156   \def\KVS@equal{%
157     \toks@\expandafter{\expandafter}\expandafter
158     \KVS@@equal\the\toks@\~\KVS@nil
159   }%
160   \def\KVS@@equal#1~#2\KVS@nil{%
161     \edef\x{\the\toks@}%
162     \ifx\x\KVS@empty
163       \toks@{#1}%
164     \else
165       \toks@\expandafter{\the\toks@=#1}%
166     \fi
167     \toks2{#2}%
168     \edef\x{\the\toks2}%
169     \ifx\x\KVS@empty
170     \else
171       \@ReturnAfterFi{%
172         \KVS@@equal#2\KVS@nil

```

```

173     }%
174     \fi
175 }%
176 }

```

\KVS@spaceA Removes one space after the equal sign. In theory also several spaces could be removed, but this is not really necessary, because T_EX usually collapses several spaces to one already.

```

177 \def\KVS@spaceA{%
178   \toks@\expandafter{\expandafter}\expandafter
179   \KVS@@spaceA\the\toks@= \KVS@nil
180 }
181 \def\KVS@@spaceA#1= #2\KVS@nil{%
182   \edef\x{\the\toks@}%
183   \ifx\x\KVS@empty
184     \toks@{#1}%
185   \else
186     \toks@\expandafter{\the\toks@=#1}%
187   \fi
188   \toks2{#2}%
189   \edef\x{\the\toks2}%
190   \ifx\x\KVS@empty
191   \else
192     \@ReturnAfterFi{%
193       \KVS@@spaceA#2\KVS@nil
194     }%
195   \fi
196 }

```

\KVS@spaceB Removes one space before the comma.

```

197 \def\KVS@spaceB#1{%
198   \toks@\expandafter{\expandafter}\expandafter
199   \KVS@@spaceB\the\toks@#1,\KVS@nil
200 }
201 \def\KVS@@spaceB#1 ,#2\KVS@nil{%
202   \edef\x{\the\toks@}%
203   \ifx\x\KVS@empty
204     \toks@{#1}%
205   \else
206     \toks@\expandafter{\the\toks@,#1}%
207   \fi
208   \toks2{#2}%
209   \edef\x{\the\toks2}%
210   \ifx\x\KVS@empty
211   \else
212     \@ReturnAfterFi{%
213       \KVS@@spaceB#2\KVS@nil
214     }%
215   \fi
216 }

```

\KVS@spaceC Removes one space after the comma.

```

217 \def\KVS@spaceC{%
218   \toks@\expandafter{\expandafter}\expandafter
219   \KVS@@spaceC\the\toks@, \KVS@nil
220 }
221 \def\KVS@@spaceC#1, #2\KVS@nil{%
222   \edef\x{\the\toks@}%
223   \ifx\x\KVS@empty
224     \toks@{#1}%
225   \else
226     \toks@\expandafter{\the\toks@,#1}%

```

```

227 \fi
228 \toks2{#2}%
229 \edef\x{\the\toks2}%
230 \ifx\x\KVS@empty
231 \else
232 \ReturnAfterFi{%
233 \KVS@@spaceC#2\KVS@nil
234 }%
235 \fi
236 }

```

`\KVS@spaceD` Removes one space before the equal sign.

```

237 \def\KVS@spaceD#1{%
238 \toks@{\expandafter{\expandafter}\expandafter
239 \KVS@@spaceD\the\toks@#1=\KVS@nil
240 }
241 \def\KVS@@spaceD#1 =#2\KVS@nil{%
242 \edef\x{\the\toks@}%
243 \ifx\x\KVS@empty
244 \toks@{#1}%
245 \else
246 \toks@{\expandafter{\the\toks@=#1}%
247 \fi
248 \toks2{#2}%
249 \edef\x{\the\toks2}%
250 \ifx\x\KVS@empty
251 \else
252 \ReturnAfterFi{%
253 \KVS@@spaceD#2\KVS@nil
254 }%
255 \fi
256 }

```

3.3 Parsing key value lists

`\kv@parse` Normalizes and parses the key value list. Also sets `\kv@list`.

```

257 \def\kv@parse#1{%
258 \kv@normalize{#1}%
259 \expandafter\kv@parse@normalized\expandafter{\kv@list}%
260 }

```

`\kv@parse@normalized`

```

261 \def\kv@parse@normalized#1#2{%
262 \KVS@parse#1,\KVS@nil{#2}%
263 }

264 \def\KVS@parse#1,#2\KVS@nil#3{%
265 \begingroup
266 \toks@{#1}%
267 \edef\x{\the\toks@}%
268 \expandafter\endgroup
269 \ifx\x\KVS@empty
270 \else
271 \KVS@process#1=\KVS@nil{#3}%
272 \fi
273 \begingroup
274 \toks@{#2}%
275 \edef\x{\the\toks@}%
276 \expandafter\endgroup
277 \ifx\x\KVS@empty
278 \else
279 \ReturnAfterFi{%

```



```

280     \KVS@parse#2\KVS@nil{#3}%
281   }%
282 \fi
283 }

284 \def\KVS@process#1=#2\KVS@nil#3{%
285   \def\kv@key{#1}%
286   \begingroup
287     \toks@{#2}%
288     \edef\x{\the\toks@}%
289   \expandafter\endgroup
290   \ifx\x\KVS@empty
291     \let\kv@value\relax
292     #3{#1}{}%
293   \else
294     \KVS@@process{#1}#2\KVS@nil{#3}%
295   \fi
296 }

297 \def\KVS@@process#1#2=\KVS@nil#3{%
298   \begingroup
299     \toks@{#2}%
300     \xdef\KVS@global{\the\toks@}%
301   \endgroup
302   \let\kv@value\KVS@global
303   #3{#1}{#2}%
304 }

```

3.4 Processing key value pairs

\kv@processor@default

```

305 \def\kv@processor@default#1#2#3{%
306   \begingroup\expandafter\expandafter\expandafter\endgroup
307   \expandafter\ifx\csname KV@#1@#2\endcsname\relax
308     \begingroup\expandafter\expandafter\expandafter\endgroup
309     \expandafter\ifx\csname KVS@#1@handler\endcsname\relax
310       \kv@error@unknownkey{#1}{#2}%
311     \else
312       \csname KVS@#1@handler\endcsname{#2}{#3}%
313     \relax
314   \fi
315 \else
316   \ifx\kv@value\relax
317     \begingroup\expandafter\expandafter\expandafter\endgroup
318     \expandafter\ifx\csname KV@#1@#2@default\endcsname\relax
319       \kv@error@novalue{#1}{#2}%
320     \else
321       \csname KV@#1@#2@default\endcsname
322     \relax
323   \fi
324 \else
325   \csname KV@#1@#2\endcsname{#3}%
326 \fi
327 \fi
328 }

```

\kv@set@family@handler

```

329 \def\kv@set@family@handler#1{%
330   \KVS@set@family@handler{#1}\@nil
331 }
332 \def\KVS@set@family@handler#1\@nil{%
333   \expandafter\def\csname KVS@#1@handler\endcsname##1##2%
334 }

```

3.5 Error handling

```
\kv@error@novalue Only a poor \PackageError is provided by miniltx.tex.
335 \expandafter\ifx\csname MessageBreak\endcsname\relax
336   \def\MessageBreak{^^J}%
337 \fi
338 \expandafter\ifx\csname @ehc\endcsname\relax
339   \def\@ehc{%
340     Try typing \space\string<return\string> %
341     \space to proceed.\MessageBreak
342     If that doesn't work, type \space X %
343     \string<return\string> \space to quit\string.%
344   }%
345 \fi
346 \def\kv@error@novalue{%
347   \kv@error@generic{No value specified for}%
348 }
349 \def\kv@error@unknownkey{%
350   \kv@error@generic{Undefined}%
351 }
352 \def\kv@error@generic#1#2#3{%
353   \begingroup
354     \newlinechar=10 %
355     \def\MessageBreak{^^J}%
356     \expandafter\ifx\csname PackageError\endcsname\relax
357       \edef\x{%
358         \errhelp{%
359           The keyval family of the key '#3' is '#2'.\MessageBreak
360           \MessageBreak
361           \@ehc
362         }%
363       }%
364       \x
365       \errmessage{kvsetkeys: #1 key '#3'}%
366     \else
367       \edef\x{%
368         \noexpand\PackageError{kvsetkeys}{%
369           #1 key '#3'%
370         }{%
371           The keyval family of the key '#3' is '#2'.\MessageBreak
372           \MessageBreak
373           \@ehc
374         }%
375       }%
376       \x
377     \fi
378   \endgroup
379 }%
```

3.6 Do it all

```
\kvsetkeys
380 \def\kvsetkeys#1#2{%
381   \kv@parse{#2}{\kv@processor@default{#1}}%
382 }
383 \KVS@endinput
384 </package>
```

4 Installation

4.1 Download

Package. This package is available on CTAN¹:

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

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

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 kvsetkeys.dtx
```

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

<code>kvsetkeys.sty</code>	→	<code>tex/generic/oberdiek/kvsetkeys.sty</code>
<code>kvsetkeys.pdf</code>	→	<code>doc/latex/oberdiek/kvsetkeys.pdf</code>
<code>kvsetkeys-example.tex</code>	→	<code>doc/latex/oberdiek/kvsetkeys-example.tex</code>
<code>kvsetkeys.dtx</code>	→	<code>source/latex/oberdiek/kvsetkeys.dtx</code>

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` , `mik \TeX` , ...) relies on file name databases, you must refresh these. For example, `te \TeX` users run `texhash` or `mktextlsr`.

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 kvsetkeys.pdf unpack_files output .
```

¹[ftp://ftp.ctan.org/tex-archive/](http://ftp.ctan.org/tex-archive/)

Unpacking with L^AT_EX. The .dtx chooses its action depending on the format:

plain-T_EX: Run docstrip and extract the files.

L^AT_EX: Generate the documentation.

If you insist on using L^AT_EX for docstrip (really, docstrip does not need L^AT_EX), then inform the autodetect routine about your intention:

```
latex \let\install=y\input{kvsetkeys.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 pdfL^AT_EX:

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

5 References

- [1] David Carlisle: *The keyval package*; 1999/03/16 v1.13; [CTAN:macros/latex/required/graphics/keyval.dtx](#).

6 History

[2006/03/06 v1.0]

- First version.

[2006/10/19 v1.1]

- Fix of \kv@set@family@handler.
- Example added.

7 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; numbers in *roman* refer to the code lines where the entry is used.

Symbols	
\,	133 , 134
\=	36 , 153 , 154
\>	38 , 39 , 40 , 41 , 42 , 43
\@ReturnAfterFi	118 , 146 , 171 , 192 , 212 , 232 , 252 , 279
\@ehc	339 , 361 , 373
\@empty	13
\@endslash	13 , 16 , 30
\@nil	330 , 332
\ \	37 , 38 , 39 , 40 , 41 , 42 , 43 , 44
\~	134 , 154
B	
\begin	34 , 35
C	
\catcode	50 , 51 , 52 , 53 , 54 , 78 , 79 , 80 , 81 , 82 , 100 ,

101, 102, 103, 104, 105, 106, 110, 111, 112, 113, 114, 115, 116	\KVS@spaceB 125, <u>197</u>
\csname 55,	\KVS@spaceC 126, <u>217</u>
65, 83, 96, 99, 307, 309, 312,	\KVS@spaceD 127, <u>237</u>
318, 321, 325, 333, 335, 338, 356	\kvsetkeys 3, 14, <u>380</u>
D	L
\define@key 29	\lccode 133, 134, 153, 154
\documentclass 2	\lowercase 135, 155
\dots 38, 41, 43	M
E	\makeatletter 7
\empty 59	\makeatother 32
\end 45, 46	\mbox 36
\endcsname 55,	\MessageBreak 336, 341, 355, 359, 360, 371, 372
65, 83, 96, 99, 307, 309, 312,	N
318, 321, 325, 333, 335, 338, 356	\newcommand 8
\endinput 74, 108	\newlinechar 354
\errhelp 358	P
\errmessage 365	\PackageError 368
I	\PackageInfo 70
\ifcase 56	\ProvidesPackage 97
\ifx ... 57, 59, 65, 83, 91, 144, 162,	Q
169, 183, 190, 203, 210, 223,	\qqquad 36
230, 243, 250, 269, 277, 290,	S
307, 309, 316, 318, 335, 338, 356	\space 25, 340, 341, 342, 343
\immediate 67, 85	T
K	\tag 8, 37, 39, 40, 42, 44
\kill 36	\textgreater 16
\kv@error@generic 347, 350, 352	\textless 16
\kv@error@novalue 319, <u>335</u>	\texttt 15
\kv@error@unknownkey 310, 349	\the 16, 24, 100, 101, 102, 103,
\kv@global 128, 130	104, 105, 106, 128, 138, 141,
\kv@key 285	143, 158, 161, 165, 168, 179,
\kv@list 130, 259	182, 186, 189, 199, 202, 206,
\kv@normalize 2, <u>119</u> , 258	209, 219, 222, 226, 229, 239,
\kv@parse 2, <u>257</u> , 381	242, 246, 249, 267, 275, 288, 300
\kv@parse@normalized 2, 259, <u>261</u>	\toks 142, 143, 167, 168, 188,
\kv@processor@default ... 3, <u>305</u> , 381	189, 208, 209, 228, 229, 248, 249
\kv@set@family@handler ... 3, 20, <u>329</u>	\toks@ 12, 16, 23, 24, 121,
\kv@value 291, 302, 316	128, 137, 138, 141, 157, 158,
\KVS@@comma 138, 140, 147	161, 163, 165, 178, 179, 182,
\KVS@@equal 158, 160, 172	184, 186, 198, 199, 202, 204,
\KVS@@process 294, 297	206, 218, 219, 222, 224, 226,
\KVS@@spaceA 179, 181, 193	238, 239, 242, 244, 246, 266,
\KVS@@spaceB 199, 201, 213	267, 274, 275, 287, 288, 299, 300
\KVS@@spaceC 219, 221, 233	U
\KVS@@spaceD 239, 241, 253	\usepackage 3, 4, 5
\KVS@comma 122, <u>132</u>	W
\KVS@empty 117, 144,	\write 67, 85
162, 169, 183, 190, 203, 210,	X
223, 230, 243, 250, 269, 277, 290	\x 55, 57,
\KVS@endinput 383	59, 66, 70, 72, 84, 89, 96, 143,
\KVS@equal 123, <u>152</u>	144, 161, 162, 168, 169, 182,
\KVS@global 300, 302	183, 189, 190, 202, 203, 209,
\KVS@nil 138, 140, 147, 158, 160, 172,	210, 222, 223, 229, 230, 242,
179, 181, 193, 199, 201, 213,	243, 249, 250, 267, 269, 275,
219, 221, 233, 239, 241, 253,	277, 288, 290, 357, 364, 367, 376
262, 264, 271, 280, 284, 294, 297	
\KVS@parse 262, 264, 280	
\KVS@process 271, 284	
\KVS@set@family@handler ... 330, 332	
\KVS@spaceA 124, <u>177</u>	