

# 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

<b>1</b>	<b>Documentation</b>	<b>1</b>
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
<b>2</b>	<b>Example</b>	<b>3</b>
<b>3</b>	<b>Implementation</b>	<b>4</b>
3.1	Identification . . . . .	4
3.2	Normalizing key value lists . . . . .	5
3.3	Parsing key value lists . . . . .	8
3.4	Processing key value pairs . . . . .	9
3.5	Error handling . . . . .	9
3.6	Do it all . . . . .	10
<b>4</b>	<b>Installation</b>	<b>10</b>
4.1	Download . . . . .	10
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
<b>5</b>	<b>References</b>	<b>12</b>
<b>6</b>	<b>History</b>	<b>12</b>
	[2006/03/06 v1.0] . . . . .	12
	[2006/10/19 v1.1] . . . . .	12
<b>7</b>	<b>Index</b>	<b>12</b>

## 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  \begingroup
12    \toks@={}%
13    \let\@endslash\@empty
14    \kvsetkeys{tag}{#1}%
15    \texttt{%
16      \textless #2\the\toks@\@endslash\textgreater
17    }%
18  \endgroup
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<sup>A</sup>T<sub>E</sub>X.

```

49 \begingroup
50   \expandafter\let\expandafter\x\csname ver@kvsetkeys.sty\endcsname
51   \ifcase 0%
52     \ifx\x\relax % plain
53   \else
54     \ifx\x\empty % LaTeX
55   \else
56     1%
57   \fi
58   \fi
59 \else

```

```

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

```

### 3.2 Normalizing key value lists

\kv@normalize

```

109 \def\kv@normalize#1{%
110 \begingroup
111 \toks@{, #1}%
112 \KVS@comma
113 \KVS@equal
114 \KVS@spaceA
115 \KVS@spaceB{ }%
116 \KVS@spaceC

```

```

117   \KVS@spaceD{ }%
118   \xdef\kv@global{\the\toks@}%
119   \endgroup
120   \let\kv@list\kv@global
121 }

```

**\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.

```

122 \begingroup
123   \lccode'\,='\",%
124   \lccode'\~='\",%
125 \lowercase{\endgroup
126   \def\KVS@comma{%
127     \toks@\expandafter{\expandafter}\expandafter
128     \KVS@@comma\the\toks@\KVS@nil
129   }%
130   \def\KVS@@comma#1~#2\KVS@nil{%
131     \toks@\expandafter{\the\toks@#1,}%
132     \toks2{#2}%
133     \edef\x{\the\toks2}%
134     \ifx\x\KVS@empty
135     \else
136       \@ReturnAfterFi{%
137         \KVS@@comma#2\KVS@nil
138       }%
139     \fi
140   }%
141 }

```

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

```

142 \begingroup
143   \lccode'\,='\",%
144   \lccode'\~='\",%
145 \lowercase{\endgroup
146   \def\KVS@equal{%
147     \toks@\expandafter{\expandafter}\expandafter
148     \KVS@@equal\the\toks@\KVS@nil
149   }%
150   \def\KVS@@equal#1~#2\KVS@nil{%
151     \edef\x{\the\toks@}%
152     \ifx\x\KVS@empty
153       \toks@{#1}%
154     \else
155       \toks@\expandafter{\the\toks@=#1}%
156     \fi
157     \toks2{#2}%
158     \edef\x{\the\toks2}%
159     \ifx\x\KVS@empty
160     \else
161       \@ReturnAfterFi{%
162         \KVS@@equal#2\KVS@nil
163       }%
164     \fi
165   }%
166 }

```

**\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<sub>E</sub>X usually collapses several spaces to one already.

```

167 \def\KVS@spaceA{%
168   \toks@\expandafter{\expandafter}\expandafter
169   \KVS@@spaceA\the\toks@=\KVS@nil

```

```

170 }
171 \def\KVS@@spaceA#1= #2\KVS@nil{%
172   \edef\x{\the\toks@}%
173   \ifx\x\KVS@empty
174     \toks@{#1}%
175   \else
176     \toks@\expandafter{\the\toks@=#1}%
177   \fi
178   \toks2{#2}%
179   \edef\x{\the\toks2}%
180   \ifx\x\KVS@empty
181   \else
182     \@ReturnAfterFi{%
183       \KVS@@spaceA#2\KVS@nil
184     }%
185   \fi
186 }

```

\KVS@spaceB Removes one space before the comma.

```

187 \def\KVS@spaceB#1{%
188   \toks@\expandafter{\expandafter}\expandafter
189   \KVS@@spaceB\the\toks@#1,\KVS@nil
190 }
191 \def\KVS@@spaceB#1 ,#2\KVS@nil{%
192   \edef\x{\the\toks@}%
193   \ifx\x\KVS@empty
194     \toks@{#1}%
195   \else
196     \toks@\expandafter{\the\toks@,#1}%
197   \fi
198   \toks2{#2}%
199   \edef\x{\the\toks2}%
200   \ifx\x\KVS@empty
201   \else
202     \@ReturnAfterFi{%
203       \KVS@@spaceB#2\KVS@nil
204     }%
205   \fi
206 }

```

\KVS@spaceC Removes one space after the comma.

```

207 \def\KVS@spaceC{%
208   \toks@\expandafter{\expandafter}\expandafter
209   \KVS@@spaceC\the\toks@, \KVS@nil
210 }
211 \def\KVS@@spaceC#1, #2\KVS@nil{%
212   \edef\x{\the\toks@}%
213   \ifx\x\KVS@empty
214     \toks@{#1}%
215   \else
216     \toks@\expandafter{\the\toks@,#1}%
217   \fi
218   \toks2{#2}%
219   \edef\x{\the\toks2}%
220   \ifx\x\KVS@empty
221   \else
222     \@ReturnAfterFi{%
223       \KVS@@spaceC#2\KVS@nil
224     }%
225   \fi
226 }

```

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

```

227 \def\KVS@spaceD#1{%
228   \toks@{\expandafter{\expandafter}\expandafter
229     \KVS@@@spaceD\the\toks@#1=\KVS@nil
230 }
231 \def\KVS@@@spaceD#1 =#2\KVS@nil{%
232   \edef\x{\the\toks@}%
233   \ifx\x\KVS@empty
234     \toks@{#1}%
235   \else
236     \toks@{\expandafter{\the\toks@=#1}%
237   \fi
238   \toks2{#2}%
239   \edef\x{\the\toks2}%
240   \ifx\x\KVS@empty
241   \else
242     \@ReturnAfterFi{%
243       \KVS@@@spaceD#2\KVS@nil
244     }%
245   \fi
246 }

```

### 3.3 Parsing key value lists

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

```

247 \def\kv@parse#1{%
248   \kv@normalize{#1}%
249   \expandafter\kv@parse@normalized\expandafter{\kv@list}%
250 }

```

`\kv@parse@normalized`

```

251 \def\kv@parse@normalized#1#2{%
252   \KVS@parse#1,\KVS@nil{#2}%
253 }

254 \def\KVS@parse#1,#2\KVS@nil#3{%
255   \begingroup
256     \toks@{#1}%
257     \edef\x{\the\toks@}%
258     \expandafter\endgroup
259     \ifx\x\KVS@empty
260     \else
261       \KVS@process#1=\KVS@nil{#3}%
262     \fi
263   \begingroup
264     \toks@{#2}%
265     \edef\x{\the\toks@}%
266     \expandafter\endgroup
267     \ifx\x\KVS@empty
268     \else
269       \@ReturnAfterFi{%
270         \KVS@parse#2\KVS@nil{#3}%
271       }%
272     \fi
273 }

274 \def\KVS@process#1=#2\KVS@nil#3{%
275   \def\kv@key{#1}%
276   \begingroup
277     \toks@{#2}%
278     \edef\x{\the\toks@}%
279     \expandafter\endgroup

```



```

280 \ifx\x\KVS@empty
281 \let\kv@value\relax
282 #3{#1}{}%
283 \else
284 \KVS@@process{#1}#2\KVS@nil{#3}%
285 \fi
286 }
287 \def\KVS@@process#1#2=\KVS@nil#3{%
288 \begingroup
289 \toks@{#2}%
290 \xdef\KVS@global{\the\toks@}%
291 \endgroup
292 \let\kv@value\KVS@global
293 #3{#1}{#2}%
294 }

```

### 3.4 Processing key value pairs

\kv@processor@default

```

295 \def\kv@processor@default#1#2#3{%
296 \begingroup\expandafter\expandafter\expandafter\endgroup
297 \expandafter\ifx\csname KV@#1@#2\endcsname\relax
298 \begingroup\expandafter\expandafter\expandafter\endgroup
299 \expandafter\ifx\csname KVS@#1@handler\endcsname\relax
300 \kv@error@unknownkey{#1}{#2}%
301 \else
302 \csname KVS@#1@handler\endcsname{#2}{#3}%
303 \relax
304 \fi
305 \else
306 \ifx\kv@value\relax
307 \begingroup\expandafter\expandafter\expandafter\endgroup
308 \expandafter\ifx\csname KV@#1@#2@default\endcsname\relax
309 \kv@error@novalue{#1}{#2}%
310 \else
311 \csname KV@#1@#2@default\endcsname
312 \relax
313 \fi
314 \else
315 \csname KV@#1@#2\endcsname{#3}%
316 \fi
317 \fi
318 }

```

\kv@set@family@handler

```

319 \def\kv@set@family@handler#1{%
320 \KVS@set@family@handler{#1}\@nil
321 }
322 \def\KVS@set@family@handler#1\@nil{%
323 \expandafter\def\csname KVS@#1@handler\endcsname##1##2%
324 }

```

### 3.5 Error handling

\kv@error@novalue Only a poor \PackageError is provided by miniltx.tex.

```

325 \expandafter\ifx\csname MessageBreak\endcsname\relax
326 \def\MessageBreak{^^J}%
327 \fi
328 \expandafter\ifx\csname @ehc\endcsname\relax
329 \def\@ehc{%
330 Try typing \space\string<return\string> %

```



**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):

kvsetkeys.sty	→	tex/generic/oberdiek/kvsetkeys.sty
kvsetkeys.pdf	→	doc/latex/oberdiek/kvsetkeys.pdf
kvsetkeys-example.tex	→	doc/latex/oberdiek/kvsetkeys-example.tex
kvsetkeys.dtx	→	source/latex/oberdiek/kvsetkeys.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 (TeX, MikTeX, ...) relies on file name databases, you must refresh these. For example, 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 kvsetkeys.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{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<sup>A</sup>T<sub>E</sub>X:

```
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		D	
<code>\,</code>	123, 124	<code>\define@key</code>	29
<code>\=</code>	36, 143, 144	<code>\documentclass</code>	2
<code>\&gt;</code>	38, 39, 40, 41, 42, 43	<code>\dots</code>	38, 41, 43
<code>\@ReturnAfterFi</code>	108,	<b>E</b>	
	136, 161, 182, 202, 222, 242, 269	<code>\empty</code>	54
<code>\@ehc</code>	329, 351, 363	<code>\end</code>	45, 46
<code>\@empty</code>	13	<code>\endcsname</code>	50,
<code>\@endslash</code>	13, 16, 30		60, 73, 86, 89, 297, 299, 302,
<code>\@nil</code>	320, 322		308, 311, 315, 323, 325, 328, 346
<code>\@</code>	37, 38, 39, 40, 41, 42, 43, 44	<code>\endinput</code>	69, 98
<code>\~</code>	124, 144	<code>\errhelp</code>	348
<b>B</b>		<code>\errmessage</code>	355
<code>\begin</code>	34, 55	<b>I</b>	
<b>C</b>		<code>\ifcase</code>	51
<code>\catcode</code>	90, 91, 92, 93, 94, 95,	<code>\ifx</code>	52, 54, 60, 73, 81, 134, 152,
	96, 100, 101, 102, 103, 104, 105, 106		159, 173, 180, 193, 200, 213,
<code>\csname</code>	50,		220, 233, 240, 259, 267, 280,
	60, 73, 86, 89, 297, 299, 302,		297, 299, 306, 308, 325, 328, 346
	308, 311, 315, 323, 325, 328, 346	<code>\immediate</code>	62, 75

<b>K</b>		<b>N</b>	
\kill	36	\newcommand	8
\kv@error@generic	337, 340, 342	\newlinechar	344
\kv@error@novalue	309, 325	<b>P</b>	
\kv@error@unknownkey	300, 339	\PackageError	358
\kv@global	118, 120	\PackageInfo	65
\kv@key	275	\ProvidesPackage	87
\kv@list	120, 249	<b>Q</b>	
\kv@normalize	2, 109, 248	\qqquad	36
\kv@parse	2, 247, 371	<b>S</b>	
\kv@parse@normalized	2, 249, 251	\space	25, 330, 331, 332, 333
\kv@processor@default	3, 295, 371	<b>T</b>	
\kv@set@family@handler	3, 20, 319	\tag	8, 37, 39, 40, 42, 44
\kv@value	281, 292, 306	\textgreater	16
\KVS@@comma	128, 130, 137	\textless	16
\KVS@@equal	148, 150, 162	\texttt	15
\KVS@@process	284, 287	\the	16, 24, 90, 91, 92, 93, 94, 95, 96, 118, 128, 131, 133, 148, 151, 155, 158, 169, 172, 176, 179, 189, 192, 196, 199, 209, 212, 216, 219, 229, 232, 236, 239, 257, 265, 278, 290
\KVS@@spaceA	169, 171, 183	\toks	132, 133, 157, 158, 178, 179, 198, 199, 218, 219, 238, 239
\KVS@@spaceB	189, 191, 203	\toks@	12, 16, 23, 24, 111, 118, 127, 128, 131, 147, 148, 151, 153, 155, 168, 169, 172, 174, 176, 188, 189, 192, 194, 196, 208, 209, 212, 214, 216, 228, 229, 232, 234, 236, 256, 257, 264, 265, 277, 278, 289, 290
\KVS@@spaceC	209, 211, 223	<b>U</b>	
\KVS@@spaceD	229, 231, 243	\usepackage	3, 4, 5
\KVS@comma	112, 122	<b>W</b>	
\KVS@empty	107, 134, 152, 159, 173, 180, 193, 200, 213, 220, 233, 240, 259, 267, 280	\write	62, 75
\KVS@endinput	373	<b>X</b>	
\KVS@equal	113, 142	\x	50, 52, 54, 61, 65, 67, 74, 79, 86, 133, 134, 151, 152, 158, 159, 172, 173, 179, 180, 192, 193, 199, 200, 212, 213, 219, 220, 232, 233, 239, 240, 257, 259, 265, 267, 278, 280, 347, 354, 357, 366
\KVS@global	290, 292	<b>L</b>	
\KVS@nil	128, 130, 137, 148, 150, 162, 169, 171, 183, 189, 191, 203, 209, 211, 223, 229, 231, 243, 252, 254, 261, 270, 274, 284, 287	\lccode	123, 124, 143, 144
\KVS@parse	252, 254, 270	\lowercase	125, 145
\KVS@process	261, 274	<b>M</b>	
\KVS@set@family@handler	320, 322	\makeatletter	7
\KVS@spaceA	114, 167	\makeatother	32
\KVS@spaceB	115, 187	\mbox	36
\KVS@spaceC	116, 207	\MessageBreak	326, 331, 345, 349, 350, 361, 362
\KVS@spaceD	117, 227	<b>L</b>	
\kvsetkeys	3, 14, 370	<b>X</b>	