

The kvsetkeys package

Heiko Oberdiek
<oberdiek@uni-freiburg.de>

2006/03/06 v1.0

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 are removed from the values.

Contents

1	Documentation	1
1.1	Normalizing key value lists	1
1.2	Parsing key value lists	2
1.3	Processing key value pairs	3
1.4	Do it all	3
2	Implementation	3
2.1	Identification	3
2.2	Normalizing key value lists	4
2.3	Parsing key value lists	7
2.4	Processing key value pairs	8
2.5	Error handling	8
2.6	Do it all	9
3	Installation	9
3.1	Some details for the interested	10
4	References	10
5	History	10
	[2006/03/06 v1.0]	10
6	Index	10

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

```

`\kv@set@family@handler {<family>} {<handler>}`

This sets the handler `<handler>` for the keyval family `<family>`.

1.4 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>}}
```

2 Implementation

2.1 Identification

```
1 <*package>
```

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

```

2 \begingroup
3   \expandafter\let\expandafter\x\csname ver@kvsetkeys.sty\endcsname
4   \ifcase 0%
5     \ifx\x\relax % plain
6     \else
7       \ifx\x\empty % LaTeX
8       \else
9         1%
10        \fi
11       \fi
12   \else
13     \expandafter\ifx\csname PackageInfo\endcsname\relax
14       \def\x#1#2{%
15         \immediate\write-1{Package #1 Info: #2.}%
16       }%
17     \else
18       \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
19     \fi
20   \x{kvsetkeys}{The package is already loaded}%
21 \endgroup

```

```

22 \expandafter\endinput
23 \fi
24 \endgroup
Package identification:
25 \begingroup
26 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
27 \def\x#1#2#3[#4]{\endgroup
28 \immediate\write-1{Package: #3 #4}%
29 \xdef#1{#4}%
30 }%
31 \else
32 \def\x#1#2[#3]{\endgroup
33 #2[#{#3}]%
34 \ifx#1\relax
35 \xdef#1{#3}%
36 \fi
37 }%
38 \fi
39 \expandafter\x\csname ver@kvsetkeys.sty\endcsname
40 \ProvidesPackage{kvsetkeys}%
41 [2006/03/06 v1.0 Key value parser with default handler support (H0)]
42 \expandafter\edef\csname KVS@endinput\endcsname{%
43 \catcode39 \the\catcode39 % '
44 \catcode44 \the\catcode44 % ,
45 \catcode61 \the\catcode61 % =
46 \catcode64 \the\catcode64 % @
47 \catcode94 \the\catcode94 % ^
48 \catcode96 \the\catcode96 % '
49 \catcode126 \the\catcode126 % ~
50 \relax
51 \noexpand\endinput
52 }
53 \catcode39 12 % '
54 \catcode44 12 % ,
55 \catcode61 12 % =
56 \catcode64 11 % @
57 \catcode94 7 % ^
58 \catcode96 12 % '
59 \catcode126 13 % ~
60 \def\KVS@empty{}
61 \long\def\@ReturnAfterFi#1\fi{\fi#1}

```

2.2 Normalizing key value lists

`\kv@normalize`

```

62 \def\kv@normalize#1{%
63 \begingroup
64 \toks@{, #1}%
65 \KVS@comma
66 \KVS@equal
67 \KVS@spaceA
68 \KVS@spaceB{ }%
69 \KVS@spaceC
70 \KVS@spaceD{ }%
71 \xdef\kv@global{\the\toks@}%
72 \endgroup
73 \let\kv@list\kv@global
74 }

```

`\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.

```

75 \begingroup

```

```

76 \lccode'\,='\",%
77 \lccode'\~='\",%
78 \lowercase{\endgroup
79 \def\KVS@comma{%
80 \toks@\expandafter{\expandafter}\expandafter
81 \KVS@@comma\the\toks@\KVS@nil
82 }
83 \def\KVS@@comma#1~#2\KVS@nil{%
84 \toks@\expandafter{\the\toks@#1,}%
85 \toks2{#2}%
86 \edef\x{\the\toks2}%
87 \ifx\x\KVS@empty
88 \else
89 \@ReturnAfterFi{%
90 \KVS@@comma#2\KVS@nil
91 }%
92 \fi
93 }%
94 }

```

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

```

95 \begingroup
96 \lccode'\=='\,%
97 \lccode'\~='\",%
98 \lowercase{\endgroup
99 \def\KVS@equal{%
100 \toks@\expandafter{\expandafter}\expandafter
101 \KVS@@equal\the\toks@\KVS@nil
102 }
103 \def\KVS@@equal#1~#2\KVS@nil{%
104 \edef\x{\the\toks@}%
105 \ifx\x\KVS@empty
106 \toks@{#1}%
107 \else
108 \toks@\expandafter{\the\toks@=#1}%
109 \fi
110 \toks2{#2}%
111 \edef\x{\the\toks2}%
112 \ifx\x\KVS@empty
113 \else
114 \@ReturnAfterFi{%
115 \KVS@@equal#2\KVS@nil
116 }%
117 \fi
118 }%
119 }

```

`\KVS@spaceA` Removes one space after the equal sign. In theory also several spaces could be removed, but this is not really necessary, because $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ usually collapses several spaces to one already.

```

120 \def\KVS@spaceA{%
121 \toks@\expandafter{\expandafter}\expandafter
122 \KVS@@spaceA\the\toks@=\KVS@nil
123 }
124 \def\KVS@@spaceA#1= #2\KVS@nil{%
125 \edef\x{\the\toks@}%
126 \ifx\x\KVS@empty
127 \toks@{#1}%
128 \else
129 \toks@\expandafter{\the\toks@=#1}%
130 \fi
131 \toks2{#2}%

```

```

132 \edef\x{\the\toks2}%
133 \ifx\x\KVS@empty
134 \else
135 \ReturnAfterFi{%
136 \KVS@@spaceA#2\KVS@nil
137 }%
138 \fi
139 }

```

\KVS@spaceB Removes one space before the comma.

```

140 \def\KVS@spaceB#1{%
141 \toks@\expandafter{\expandafter}\expandafter
142 \KVS@@spaceB\the\toks@#1,\KVS@nil
143 }
144 \def\KVS@@spaceB#1 ,#2\KVS@nil{%
145 \edef\x{\the\toks@}%
146 \ifx\x\KVS@empty
147 \toks@{#1}%
148 \else
149 \toks@\expandafter{\the\toks@,#1}%
150 \fi
151 \toks2{#2}%
152 \edef\x{\the\toks2}%
153 \ifx\x\KVS@empty
154 \else
155 \ReturnAfterFi{%
156 \KVS@@spaceB#2\KVS@nil
157 }%
158 \fi
159 }

```

\KVS@spaceC Removes one space after the comma.

```

160 \def\KVS@spaceC{%
161 \toks@\expandafter{\expandafter}\expandafter
162 \KVS@@spaceC\the\toks@, \KVS@nil
163 }
164 \def\KVS@@spaceC#1 ,#2\KVS@nil{%
165 \edef\x{\the\toks@}%
166 \ifx\x\KVS@empty
167 \toks@{#1}%
168 \else
169 \toks@\expandafter{\the\toks@,#1}%
170 \fi
171 \toks2{#2}%
172 \edef\x{\the\toks2}%
173 \ifx\x\KVS@empty
174 \else
175 \ReturnAfterFi{%
176 \KVS@@spaceC#2\KVS@nil
177 }%
178 \fi
179 }

```

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

```

180 \def\KVS@spaceD#1{%
181 \toks@\expandafter{\expandafter}\expandafter
182 \KVS@@spaceD\the\toks@#1=\KVS@nil
183 }
184 \def\KVS@@spaceD#1 =#2\KVS@nil{%
185 \edef\x{\the\toks@}%
186 \ifx\x\KVS@empty
187 \toks@{#1}%

```

```

188 \else
189   \toks@\expandafter{\the\toks@=#1}%
190 \fi
191 \toks2{#2}%
192 \edef\x{\the\toks2}%
193 \ifx\x\KVS@empty
194 \else
195   \@ReturnAfterFi{%
196     \KVS@@spaceD#2\KVS@nil
197   }%
198 \fi
199 }

```

2.3 Parsing key value lists

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

```

200 \def\kv@parse#1{%
201   \kv@normalize{#1}%
202   \expandafter\kv@parse@normalized\expandafter{\kv@list}%
203 }

```

`\kv@parse@normalized`

```

204 \def\kv@parse@normalized#1#2{%
205   \KVS@parse#1,\KVS@nil{#2}%
206 }

207 \def\KVS@parse#1,#2\KVS@nil#3{%
208   \begingroup
209     \toks@{#1}%
210     \edef\x{\the\toks@}%
211     \expandafter\endgroup
212     \ifx\x\KVS@empty
213     \else
214       \KVS@process#1=\KVS@nil{#3}%
215     \fi
216   \begingroup
217     \toks@{#2}%
218     \edef\x{\the\toks@}%
219     \expandafter\endgroup
220     \ifx\x\KVS@empty
221     \else
222       \@ReturnAfterFi{%
223         \KVS@parse#2\KVS@nil{#3}%
224       }%
225     \fi
226 }

227 \def\KVS@process#1=#2\KVS@nil#3{%
228   \def\kv@key{#1}%
229   \begingroup
230     \toks@{#2}%
231     \edef\x{\the\toks@}%
232     \expandafter\endgroup
233     \ifx\x\KVS@empty
234       \let\kv@value\relax
235       #3{#1}-{ }%
236     \else
237       \KVS@@process{#1}#2\KVS@nil{#3}%
238     \fi
239 }

240 \def\KVS@@process#1#2=\KVS@nil#3{%
241   \begingroup
242     \toks@{#2}%

```

```

243 \xdef\KVS@global{\the\toks@}%
244 \endgroup
245 \let\kv@value\KVS@global
246 #3{#1}{#2}%
247 }

```

2.4 Processing key value pairs

`\kv@processor@default`

```

248 \def\kv@processor@default#1#2#3{%
249 \begingroup\expandafter\expandafter\expandafter\endgroup
250 \expandafter\ifx\csname KV@#1@#2\endcsname\relax
251 \begingroup\expandafter\expandafter\expandafter\endgroup
252 \expandafter\ifx\csname KVS@#1@handler\endcsname\relax
253 \kv@error@unknownkey{#1}{#2}%
254 \else
255 \csname KVS@#1@handler\endcsname{#2}{#3}%
256 \relax
257 \fi
258 \else
259 \ifx\kv@value\relax
260 \begingroup\expandafter\expandafter\expandafter\endgroup
261 \expandafter\ifx\csname KV@#1@#2@default\endcsname\relax
262 \kv@error@novalue{#1}{#2}%
263 \else
264 \csname KV@#1@#2@default\endcsname
265 \relax
266 \fi
267 \else
268 \csname KV@#1@#2\endcsname{#3}%
269 \fi
270 \fi
271 }

```

`\kv@set@family@handler`

```

272 \def\kv@set@family@handler#1#{%
273 \expandafter\def\csname KVS@#1@handler\endcsname
274 }

```

2.5 Error handling

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

```

275 \expandafter\ifx\csname MessageBreak\endcsname\relax
276 \def\MessageBreak{^^J}%
277 \fi
278 \expandafter\ifx\csname @ehc\endcsname\relax
279 \def\@ehc{%
280 Try typing \space\string<return\string> %
281 \space to proceed.\MessageBreak
282 If that doesn't work, type \space X %
283 \string<return\string> \space to quit\string.%
284 }%
285 \fi
286 \def\kv@error@novalue{%
287 \kv@error@generic{No value specified for}%
288 }
289 \def\kv@error@unknownkey{%
290 \kv@error@generic{Undefined}%
291 }
292 \def\kv@error@generic#1#2#3{%
293 \begingroup

```



```

294 \newlinechar=10 %
295 \def\MessageBreak{^~J}%
296 \expandafter\ifx\cename PackageError\endcename\relax
297 \edef\x{%
298 \errhelp{%
299 The keyval family of the key ‘#3’ is ‘#2’.\MessageBreak
300 \MessageBreak
301 \@ehc
302 }%
303 }%
304 \x
305 \errmessage{kvsetkeys: #1 key ‘#3’}%
306 \else
307 \edef\x{%
308 \noexpand\PackageError{kvsetkeys}{%
309 #1 key ‘#3’%
310 }{%
311 The keyval family of the key ‘#3’ is ‘#2’.\MessageBreak
312 \MessageBreak
313 \@ehc
314 }%
315 }%
316 \x
317 \fi
318 \endgroup
319 }%

```

2.6 Do it all

`\kvsetkeys`

```

320 \def\kvsetkeys#1#2{%
321 \kv@parse{#2}{\kv@processor@default{#1}}%
322 }

323 \KVS@endinput
324 </package>

```

3 Installation

CTAN. This package is available on CTAN¹:

[CTAN:macros/latex/contrib/oberdiek/kvsetkeys.dtx](http://ftp.ctan.org/tex-archive/macros/latex/contrib/oberdiek/kvsetkeys.dtx) The source file.

[CTAN:macros/latex/contrib/oberdiek/kvsetkeys.pdf](http://ftp.ctan.org/tex-archive/macros/latex/contrib/oberdiek/kvsetkeys.pdf) Documentation.

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.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.

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

Refresh file databases. If your T_EX distribution (teT_EX, mikT_EX, ...) rely on file databases, you must refresh these. For example, teT_EX users run `texhash` or `mktextlsr`.

3.1 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 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 intension:

```
latex \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
```

4 References

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

5 History

[2006/03/06 v1.0]

- First version.

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

Symbols	<code>\=</code>	96, 97
<code>\,</code>	76, 77	<code>\@ReturnAfterFi</code> 61,

89, 114, 135, 155, 175, 195, 222	162, 164, 176, 182, 184, 196,
\@ehc 279, 301, 313	205, 207, 214, 223, 227, 237, 240
\~ 77, 97	\KVS@parse 205, 207, 223
C	\KVS@process 214, 227
\catcode 43, 44, 45, 46,	\KVS@spaceA 67, 120
47, 48, 49, 53, 54, 55, 56, 57, 58, 59	\KVS@spaceB 68, 140
\csname 3, 13, 26, 39, 42, 250, 252, 255,	\KVS@spaceC 69, 160
261, 264, 268, 273, 275, 278, 296	\KVS@spaceD 70, 180
E	\kvsetkeys 3, 320
\empty 7	L
\endcsname 3,	\lccode 76, 77, 96, 97
13, 26, 39, 42, 250, 252, 255,	\lowercase 78, 98
261, 264, 268, 273, 275, 278, 296	M
\endinput 22, 51	\MessageBreak 276, 281, 295, 299, 300, 311, 312
\errhelp 298	N
\errmessage 305	\newlinechar 294
I	P
\ifcase 4	\PackageError 308
\ifx 5, 7, 13, 26, 34, 87, 105,	\PackageInfo 18
112, 126, 133, 146, 153, 166,	\ProvidesPackage 40
173, 186, 193, 212, 220, 233,	S
250, 252, 259, 261, 275, 278, 296	\space 280, 281, 282, 283
\immediate 15, 28	T
K	\the 43, 44, 45, 46, 47, 48, 49, 71, 81,
\kv@error@generic 287, 290, 292	84, 86, 101, 104, 108, 111, 122,
\kv@error@novalue 262, 275	125, 129, 132, 142, 145, 149,
\kv@error@unknownkey 253, 289	152, 162, 165, 169, 172, 182,
\kv@global 71, 73	185, 189, 192, 210, 218, 231, 243
\kv@key 228	\toks 85, 86, 110, 111, 131,
\kv@list 73, 202	132, 151, 152, 171, 172, 191, 192
\kv@normalize 1, 62, 201	\toks@ . 64, 71, 80, 81, 84, 100, 101,
\kv@parse 2, 200, 321	104, 106, 108, 121, 122, 125,
\kv@parse@normalized 2, 202, 204	127, 129, 141, 142, 145, 147,
\kv@processor@default ... 3, 248, 321	149, 161, 162, 165, 167, 169,
\kv@set@family@handler 3, 272	181, 182, 185, 187, 189, 209,
\kv@value 234, 245, 259	210, 217, 218, 230, 231, 242, 243
\KVS@@comma 81, 83, 90	W
\KVS@@equal 101, 103, 115	\write 15, 28
\KVS@@process 237, 240	X
\KVS@@spaceA 122, 124, 136	\x ... 3, 5, 7, 14, 18, 20, 27, 32, 39,
\KVS@@spaceB 142, 144, 156	86, 87, 104, 105, 111, 112, 125,
\KVS@@spaceC 162, 164, 176	126, 132, 133, 145, 146, 152,
\KVS@@spaceD 182, 184, 196	153, 165, 166, 172, 173, 185,
\KVS@comma 65, 75	186, 192, 193, 210, 212, 218,
\KVS@empty 60,	220, 231, 233, 297, 304, 307, 316
87, 105, 112, 126, 133, 146, 153,	
166, 173, 186, 193, 212, 220, 233	
\KVS@endinput 323	
\KVS@equal 66, 95	
\KVS@global 243, 245	
\KVS@nil . 81, 83, 90, 101, 103, 115,	
122, 124, 136, 142, 144, 156,	