

The kvsetkeys package

Heiko Oberdiek
<oberdiek@uni-freiburg.de>

2009/12/22 v1.7

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	2
1.1	Motivation	2
1.2	Normalizing key value lists	3
1.3	Parsing key value lists	3
1.4	Processing key value pairs	4
1.5	Default family handler	4
1.6	Put it all together	4
1.7	Comma separated lists	5
2	Example	5
3	Implementation	6
3.1	Identification	6
3.2	Package loading	7
3.3	Check for ε -TeX	8
3.4	Generic help macros	8
3.5	Normalizing	8
3.6	Parsing key value lists	11
3.7	Parsing comma lists	12
3.8	Processing key value pairs	12
3.9	Error handling	13
3.10	Do it all	13
4	Test	13
4.1	Catcode checks for loading	13
4.2	Macro tests	15
4.2.1	Preamble	15
4.2.2	Time	15
4.2.3	Test sets	16
5	Installation	19
5.1	Download	19
5.2	Bundle installation	19
5.3	Package installation	19
5.4	Refresh file name databases	20
5.5	Some details for the interested	20

6	References	20
7	History	21
	[2006/03/06 v1.0]	21
	[2006/10/19 v1.1]	21
	[2007/09/09 v1.2]	21
	[2007/09/29 v1.3]	21
	[2009/07/19 v1.4]	21
	[2009/07/30 v1.5]	21
	[2009/12/12 v1.6]	21
	[2009/12/22 v1.7]	21
8	Index	21

1 Documentation

First I want to recommend the very good review article “A guide to key-value methods” by Joseph Wright [1]. It introduces the different key-value packages and compares them.

1.1 Motivation

`\kvsetkeys` serves as replacement for `keyval`’s `\setkeys`. It basically uses the same syntax. But the implementation is more robust and predictable:

Active syntax characters: Comma ‘,’ and the equals sign ‘=’ are used inside key value lists as syntax characters. Package `keyval` uses the catcode of the characters that is active during package loading, usually this is catcode 12 (other). But it can happen that the catcode setting of the syntax characters changes. Especially active characters are of interest, because some language adaptations uses them. For example, option `turkish` of package `babel` uses the equals sign as active shorthand character. Therefore package `kvsetkeys` deals with both catcode settings 12 (other) and 13 (active).

Brace removal: Package `keyval`’s `\setkeys` removes up to two levels of curly braces around the value in some unpredictable way:

```
\setkeys{fam}{key={{value}}}\setkeys{fam}{key={{value}}}\setkeys{fam}{key= {{{value}}}}\setkeys{fam}{key= {{{value}}}}
```

This package `kvsetkeys` follows a much stronger rule: Exactly one level of braces are removed from an item, if the item is surrounded by curly braces. An item can be a the key value pair, the key or the value.

```
\kvsetkeys{fam}{key={value}}\kvsetkeys{fam}{key={{value}}}\kvsetkeys{fam}{key= {{value}}}
```

Arbitrary values: Unmatched conditionals are supported.

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.

1.2 Normalizing key value lists

`\kv@normalize {⟨key value list⟩}`

If the user specifies key value lists, he usually prefers 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={...},...,
```

Curly braces around values (or keys) remain untouched.

v1.3+: One comma is added in front of the list and each pair ends with a comma. Thus an empty list consists of one comma, otherwise two commas encloses the list. Empty entries other than the first are removed.

v1.0 – v1.2: 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.

1.3 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}
```

1.4 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
```

1.5 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`.

1.6 Put it all together

`\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
```

1.7 Comma separated lists

Since version 2007/09/29 v1.3 this package also supports the normalizing and parsing of general comma separated lists.

`\comma@normalize{\langle comma list \rangle}`

Macro `\comma@normalize` normalizes the comma separated list, removes spaces around commas. The result is put in macro `\comma@list`.

`\comma@parse{\langle comma list \rangle}{\langle processor \rangle}`

Macro `\comma@parse` first normalizes the comma separated list and then parses the list by calling `\comma@parse@normalized`.

`\comma@parse@normalized{\langle normalized comma list \rangle}{\langle processor \rangle}`

The list is parsed. Empty entries are ignored. `\langle processor \rangle` is called for each non-empty entry with the entry as argument:

`\langle processor \rangle{\langle entry \rangle}`

Also the entry is stored in the macro `\comma@entry`.

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^AT_EX.

```

49 \begingroup
50   \catcode44 12 % ,
51   \catcode45 12 % -
52   \catcode46 12 % .
53   \catcode58 12 % :
54   \catcode64 11 % @
55   \catcode123 1 % {
56   \catcode125 2 % }
57   \expandafter\let\expandafter\x\csname ver@kvsetkeys.sty\endcsname
58   \ifx\x\relax % plain-TeX, first loading
59   \else
60     \def\empty{}%
61     \ifx\x\empty % LaTeX, first loading,
62       % variable is initialized, but \ProvidesPackage not yet seen
63     \else
64       \catcode35 6 % #
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     \aftergroup\endinput
74   \fi
75 \fi
76 \endgroup

```

Package identification:

```

77 \begingroup
78   \catcode35 6 % #
79   \catcode40 12 % (
80   \catcode41 12 % )
81   \catcode44 12 % ,
82   \catcode45 12 % -
83   \catcode46 12 % .
84   \catcode47 12 % /

```

```

85 \catcode58 12 % :
86 \catcode64 11 % @
87 \catcode91 12 % [
88 \catcode93 12 % ]
89 \catcode123 1 % {
90 \catcode125 2 % }
91 \expandafter\ifx\csname ProvidesPackage\endcsname\relax
92   \def\x#1#2#3[#4]{\endgroup
93     \immediate\write-1{Package: #3 #4}%
94     \xdef#1{#4}%
95   }%
96 \else
97   \def\x#1#2[#3]{\endgroup
98     #2[#3]}%
99   \ifx#1\@undefined
100     \xdef#1{#3}%
101   \fi
102   \ifx#1\relax
103     \xdef#1{#3}%
104   \fi
105 }%
106 \fi
107 \expandafter\x\csname ver@kvsetkeys.sty\endcsname
108 \ProvidesPackage{kvsetkeys}%
109 [2009/12/22 v1.7 Key value parser (H0)]
110 \begingroup
111 \catcode123 1 % {
112 \catcode125 2 % }
113 \def\x{\endgroup
114   \expandafter\edef\csname KVS@AtEnd\endcsname{%
115     \catcode35 \the\catcode35\relax
116     \catcode64 \the\catcode64\relax
117     \catcode123 \the\catcode123\relax
118     \catcode125 \the\catcode125\relax
119   }%
120 }%
121 \x
122 \catcode35 6 % #
123 \catcode64 11 % @
124 \catcode123 1 % {
125 \catcode125 2 % }
126 \def\TMP@EnsureCode#1#2{%
127   \edef\KVS@AtEnd{%
128     \KVS@AtEnd
129     \catcode#1 \the\catcode#1\relax
130   }%
131   \catcode#1 #2\relax
132 }
133 \TMP@EnsureCode{36}{3}% $
134 \TMP@EnsureCode{38}{4}% &
135 \TMP@EnsureCode{39}{12}% '
136 \TMP@EnsureCode{44}{12}% ,
137 \TMP@EnsureCode{46}{12}% .
138 \TMP@EnsureCode{47}{12}% /
139 \TMP@EnsureCode{61}{12}% =
140 \TMP@EnsureCode{94}{7}% ^ (superscript)
141 \TMP@EnsureCode{96}{12}% '
142 \TMP@EnsureCode{126}{13}% ~ (active)

```

3.2 Package loading

```

143 \begingroup\expandafter\expandafter\expandafter\endgroup
144 \expandafter\ifx\csname RequirePackage\endcsname\relax

```

```

145 \input infwarerr.sty\relax
146 \input etexcmds.sty\relax
147 \else
148 \RequirePackage{infwarerr}[2007/09/09]%
149 \RequirePackage{etexcmds}[2007/09/09]%
150 \fi

```

3.3 Check for ε -TeX

`\unexpanded`, `\ifcsname`, and `\unless` are used if found.

```

151 \begingroup\expandafter\endgroup
152 \ifcase0\ifetex@unexpanded
153     \expandafter\ifx\csname ifcsname\endcsname\relax
154     \else
155     \expandafter\ifx\csname unless\endcsname\relax
156     \else
157     1%
158     \fi
159     \fi
160     \fi
161 \catcode'\$=9 % ignore
162 \catcode'\&=14 % comment
163 \else % e-TeX
164 \catcode'\$=14 % comment
165 \catcode'\&=9 % ignore
166 \fi

```

3.4 Generic help macros

`\KVS@Empty`

```
167 \def\KVS@Empty{}
```

`\KVS@FirstOfTwo`

```
168 \long\def\KVS@FirstOfTwo#1#2{#1}
```

`\KVS@SecondOfTwo`

```
169 \long\def\KVS@SecondOfTwo#1#2{#2}
```

`\KVS@IfEmpty`

```

170 \def\KVS@IfEmpty#1{%
171 & \edef\KVS@Temp{\etex@unexpanded{#1}}%
172 $ \begingroup
173 $ \toks@{#1}%
174 $ \edef\KVS@Temp{\the\toks@}%
175 $ \expandafter\endgroup
176 \ifx\KVS@Temp\KVS@Empty
177 \expandafter\KVS@FirstOfTwo
178 \else
179 \expandafter\KVS@SecondOfTwo
180 \fi
181 }

```

3.5 Normalizing

`\kv@normalize`

```

182 \def\kv@normalize#1{%
183 \begingroup
184 \toks@{, #1,}%
185 \KVS@Comma
186 \KVS@SpaceComma{ }%
187 \KVS@CommaSpace

```

```

188 \KVS@CommaComma
189 \KVS@Equals
190 \KVS@SpaceEquals{ }%
191 \KVS@EqualsSpace
192 \xdef\KVS@Global{\the\toks@}%
193 \endgroup
194 \let\kv@list\KVS@Global
195 }

```

\comma@normalize

```

196 \def\comma@normalize#1{%
197 \begingroup
198 \toks@{, #1,}%
199 \KVS@Comma
200 \KVS@SpaceComma{ }%
201 \KVS@CommaSpace
202 \KVS@CommaComma
203 \xdef\KVS@Global{\the\toks@}%
204 \endgroup
205 \let\comma@list\KVS@Global
206 }

```

\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.

```

207 \begingroup
208 \lccode'\,=\,%
209 \lccode'\~= '\,%
210 \lowercase{\endgroup
211 \def\KVS@Comma{%
212 \toks@\expandafter{\expandafter}\expandafter
213 \KVS@@Comma\the\toks@\~\KVS@Nil
214 }%
215 \def\KVS@@Comma#1~#2\KVS@Nil{%
216 \toks@\expandafter{\the\toks@#1}%
217 \KVS@IfEmpty{#2}{%
218 }{%
219 \KVS@@Comma,#2\KVS@Nil
220 }%
221 }%
222 }

```

\KVS@SpaceComma Removes spaces before the comma, may add commas at the end.

```

223 \def\KVS@SpaceComma#1{%
224 \expandafter\KVS@@SpaceComma\the\toks@#1,\KVS@Nil
225 }

```

\KVS@@SpaceComma

```

226 \def\KVS@@SpaceComma#1 ,#2\KVS@Nil{%
227 \KVS@IfEmpty{#2}{%
228 \toks@{#1}%
229 }{%
230 \toks@{#1,#2}%
231 \expandafter\KVS@@SpaceComma\the\toks@\KVS@Nil
232 }%
233 }

```

\KVS@CommaSpace Removes spaces after the comma, may add commas at the end.

```

234 \def\KVS@CommaSpace{%
235 \expandafter\KVS@@CommaSpace\the\toks@, \KVS@Nil
236 }

```

`\KVS@@CommaSpace`

```
237 \def\KVS@@CommaSpace#1, #2\KVS@Nil{%
238   \KVS@IfEmpty{#2}{%
239     \toks@{#1}%
240   }{%
241     \toks@{#1,#2}%
242     \expandafter\KVS@@CommaSpace\the\toks@\KVS@Nil
243   }%
244 }
```

`\KVS@CommaComma` Replaces multiple commas by one comma.

```
245 \def\KVS@CommaComma{%
246   \expandafter\KVS@@CommaComma\the\toks@,\KVS@Nil
247 }
```

`\KVS@@CommaComma`

```
248 \def\KVS@@CommaComma#1,,#2\KVS@Nil{%
249   \toks@{#1,#2}%
250   \KVS@IfEmpty{#2}{%
251     }{%
252     \expandafter\KVS@@CommaComma\the\toks@\KVS@Nil
253   }%
254 }
```

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

```
255 \begingroup
256   \lccode'\=='\=%
257   \lccode'\~='\=%
258 \lowercase{\endgroup
259   \def\KVS@Equals{%
260     \toks@\expandafter{\expandafter}\expandafter
261     \KVS@@Equals\the\toks@\KVS@Nil
262   }%
263   \def\KVS@@Equals#1~#2\KVS@Nil{%
264     \edef\KVS@Temp{\the\toks@}%
265     \ifx\KVS@Temp\KVS@Empty
266       \expandafter\KVS@FirstOfTwo
267     \else
268       \expandafter\KVS@SecondOfTwo
269     \fi
270     {%
271       \toks@{#1}%
272     }{%
273       \toks@\expandafter{\the\toks@=#1}%
274     }%
275     \KVS@IfEmpty{#2}{%
276       }{%
277         \KVS@@Equals#2\KVS@Nil
278       }%
279     }%
280 }
```

`\KVS@SpaceEquals` Removes spaces before the equals sign.

```
281 \def\KVS@SpaceEquals#1{%
282   \expandafter\KVS@@SpaceEquals\the\toks@#1=\KVS@Nil
283 }
```

`\KVS@@SpaceEquals`

```
284 \def\KVS@@SpaceEquals#1 =#2\KVS@Nil{%
285   \KVS@IfEmpty{#2}{%
286     \toks@{#1}%
287   }
```

```

287 }{%
288   \toks@{#1=#2}%
289   \expandafter\KVS@@SpaceEquals\the\toks@\KVS@Nil
290 }%
291 }

```

\KVS@EqualsSpace Removes spaces after the equals sign.

```

292 \def\KVS@EqualsSpace{%
293   \expandafter\KVS@@EqualsSpace\the\toks@= \KVS@Nil
294 }

```

\KVS@@EqualsSpace

```

295 \def\KVS@@EqualsSpace#1= #2\KVS@Nil{%
296   \KVS@IfEmpty{#2}{%
297     \toks@{#1}%
298   }{%
299     \toks@{#1=#2}%
300     \expandafter\KVS@@EqualsSpace\the\toks@\KVS@Nil
301   }%
302 }

```

3.6 Parsing key value lists

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

```

303 \def\kv@parse#1{%
304   \kv@normalize{#1}%
305   \expandafter\kv@parse@normalized\expandafter{\kv@list}%
306 }

```

\kv@parse@normalized #1: key value list
#2: processor

```

307 \def\kv@parse@normalized#1#2{%
308   \KVS@Parse#1,\KVS@Nil{#2}%
309 }

```

\KVS@Parse #1,#2: key value list
#3: processor

```

310 \def\KVS@Parse#1,#2\KVS@Nil#3{%
311   \KVS@IfEmpty{#1}{%
312   }{%
313     \KVS@Process#1=\KVS@Nil{#3}%
314   }%
315   \KVS@IfEmpty{#2}{%
316   }{%
317     \KVS@Parse#2\KVS@Nil{#3}%
318   }%
319 }

```

\KVS@Process #1: key
#2: value, =
#3: processor

```

320 \def\KVS@Process#1=#2\KVS@Nil#3{%
321   \def\kv@key{#1}%
322   \KVS@IfEmpty{#2}{%
323     \let\kv@value\relax
324     #3{#1}{}%
325   }{%
326     \KVS@@Process{#1}#2\KVS@Nil{#3}%
327   }%
328 }

```

```

\KVS@@Process #1: key
               #2: value
               #3: processor
329 \def\KVS@@Process#1#2=\KVS@Nil#3{%
330 & \edef\kv@value{\etex@unexpanded{#2}}%
331 $ \begingroup
332 $ \toks@{#2}%
333 $ \xdef\KVS@Global{\the\toks@}%
334 $ \endgroup
335 $ \let\kv@value\KVS@Global
336 #3{#1}{#2}%
337 }

```

3.7 Parsing comma lists

```

\comma@parse Normalizes and parses the key value list. Also sets \comma@list.
338 \def\comma@parse#1{%
339 \comma@normalize{#1}%
340 \expandafter\comma@parse@normalized\expandafter{\comma@list}%
341 }

```

```

\comma@parse@normalized #1: comma list
                        #2: processor
342 \def\comma@parse@normalized#1#2{%
343 \KVS@CommaParse#1,\KVS@Nil{#2}%
344 }

```

```

\KVS@CommaParse #1,#2: comma list
                 #3: processor
345 \def\KVS@CommaParse#1,#2\KVS@Nil#3{%
346 \KVS@IfEmpty{#1}{%
347 }{%
348 \def\comma@entry{#1}%
349 #3{#1}%
350 }%
351 \KVS@IfEmpty{#2}{%
352 }{%
353 \KVS@CommaParse#2\KVS@Nil{#3}%
354 }%
355 }

```

3.8 Processing key value pairs

```

\kv@processor@default
356 \def\kv@processor@default#1#2#3{%
357 & \unless\ifcsname KV@#1@#2\endcsname
358 $ \begingroup\expandafter\expandafter\expandafter\endgroup
359 $ \expandafter\ifx\csname KV@#1@#2\endcsname\relax
360 & \unless\ifcsname KVS@#1@handler\endcsname
361 $ \begingroup\expandafter\expandafter\expandafter\endgroup
362 $ \expandafter\ifx\csname KVS@#1@handler\endcsname\relax
363 \kv@error@unknownkey{#1}{#2}%
364 \else
365 \csname KVS@#1@handler\endcsname{#2}{#3}%
366 \relax
367 \fi
368 \else
369 \ifx\kv@value\relax
370 & \unless\ifcsname KV@#1@#2@default\endcsname
371 $ \begingroup\expandafter\expandafter\expandafter\endgroup
372 $ \expandafter\ifx\csname KV@#1@#2@default\endcsname\relax

```

```

373         \kv@error@novalue{#1}{#2}%
374     \else
375         \csname KV@#1@#2@default\endcsname
376     \relax
377 \fi
378 \else
379     \csname KV@#1@#2\endcsname{#3}%
380 \fi
381 \fi
382 }

```

\kv@set@family@handler

```

383 \def\kv@set@family@handler#1{%
384     \KVS@SetFamilyHandler{#1}\@nil
385 }

```

\KVS@SetFamilyHandler

```

386 \def\KVS@SetFamilyHandler#1\@nil#{%
387     \expandafter\def\csname KVS@#1@handler\endcsname##1##2%
388 }

```

3.9 Error handling

\kv@error@novalue

```

389 \def\kv@error@novalue{%
390     \kv@error@generic{No value specified for}%
391 }

```

\kv@error@unknownkey

```

392 \def\kv@error@unknownkey{%
393     \kv@error@generic{Undefined}%
394 }

```

\kv@error@generic

```

395 \def\kv@error@generic#1#2#3{%
396     \@PackageError{kvsetkeys}{%
397         #1 key ‘#3’%
398     }{%
399         The keyval family of the key ‘#3’ is ‘#2’.\MessageBreak
400         \MessageBreak
401         \@ehc
402     }%
403 }

```

3.10 Do it all

\kvsetkeys

```

404 \def\kvsetkeys#1#2{%
405     \kv@parse{#2}{\kv@processor@default{#1}}%
406 }

407 \KVS@AtEnd
408 \</package>

```

4 Test

4.1 Catcode checks for loading

```

409 \<test1>

```

```

410 \catcode'\{=1 %
411 \catcode'\}=2 %
412 \catcode'\#=6 %
413 \catcode'\@=11 %
414 \expandafter\ifx\csname count@\endcsname\relax
415   \countdef\count@=255 %
416 \fi
417 \expandafter\ifx\csname @gobble\endcsname\relax
418   \long\def\@gobble#1{%
419 \fi
420 \expandafter\ifx\csname @firstofone\endcsname\relax
421   \long\def\@firstofone#1{#1}%
422 \fi
423 \expandafter\ifx\csname loop\endcsname\relax
424   \expandafter\@firstofone
425 \else
426   \expandafter\@gobble
427 \fi
428 {%
429   \def\loop#1\repeat{%
430     \def\body{#1}%
431     \iterate
432   }%
433   \def\iterate{%
434     \body
435     \let\next\iterate
436   \else
437     \let\next\relax
438   \fi
439   \next
440   }%
441   \let\repeat=\fi
442 }%
443 \def\RestoreCatcodes{}
444 \count@=0 %
445 \loop
446   \edef\RestoreCatcodes{%
447     \RestoreCatcodes
448     \catcode\the\count@=\the\catcode\count@\relax
449   }%
450 \ifnum\count@<255 %
451   \advance\count@ 1 %
452 \repeat
453
454 \def\RangeCatcodeInvalid#1#2{%
455   \count@=#1\relax
456   \loop
457     \catcode\count@=15 %
458   \ifnum\count@<#2\relax
459     \advance\count@ 1 %
460   \repeat
461 }
462 \expandafter\ifx\csname LoadCommand\endcsname\relax
463 \def\LoadCommand{\input kvsetkeys.sty\relax}%
464 \fi
465 \def\Test{%
466   \RangeCatcodeInvalid{0}{47}%
467   \RangeCatcodeInvalid{58}{64}%
468   \RangeCatcodeInvalid{91}{96}%
469   \RangeCatcodeInvalid{123}{255}%
470   \catcode'\@=12 %
471   \catcode'\=0 %

```

```

472 \catcode'\{=1 %
473 \catcode'\}=2 %
474 \catcode'\#=6 %
475 \catcode'\[=12 %
476 \catcode'\]=12 %
477 \catcode'\%=14 %
478 \catcode'\ =10 %
479 \catcode\l3=5 %
480 \LoadCommand
481 \RestoreCatcodes
482 }
483 \Test
484 \csname @@end\endcsname
485 \end
486 </test1>

```

4.2 Macro tests

4.2.1 Preamble

```

487 <*test2>
488 \NeedsTeXFormat{LaTeX2e}
489 \nofiles
490 \documentclass{article}
491 \noetex\let\SavedUnexpanded\unexpanded
492 \noetex\let\unexpanded\UNDEFINED
493 \makeatletter
494 \chardef\KVS@TestMode=1 %
495 \makeatother
496 \usepackage{kvsetkeys}[2009/12/22]
497 \noetex\let\unexpanded\SavedUnexpanded
498 \usepackage{qstest}
499 \IncludeTests{*}
500 \LogTests{log}{*}{*}

```

4.2.2 Time

```

501 \begingroup\expandafter\expandafter\expandafter\endgroup
502 \expandafter\ifx\csname pdfresettimer\endcsname\relax
503 \else
504   \makeatletter
505   \newcount\SummaryTime
506   \newcount\TestTime
507   \SummaryTime=\z@
508   \newcommand*\PrintTime}[2]{%
509     \typeout{%
510       [Time #1: \strip@pt\dimexpr\number#2sp\relax\space s]%
511     }%
512   }%
513   \newcommand*\StartTime#[1]{%
514     \renewcommand*\TimeDescription{#1}%
515     \pdfresettimer
516   }%
517   \newcommand*\TimeDescription{}%
518   \newcommand*\StopTime{%
519     \TestTime=\pdfelapsedtime
520     \global\advance\SummaryTime\TestTime
521     \PrintTime\TimeDescription\TestTime
522   }%
523   \let\saved@qstest\qstest
524   \let\saved@endqstest\endqstest
525   \def\qstest#1#2{%
526     \saved@qstest{#1}{#2}%
527     \StartTime{#1}%

```

```

528 }%
529 \def\endqstest{%
530     \StopTime
531     \saved@endqstest
532 }%
533 \AtEndDocument{%
534     \PrintTime{summary}\SummaryTime
535 }%
536 \makeatother
537 \fi

```

4.2.3 Test sets

```

538 \makeatletter
539 \def\@makeactive#1{%
540     \catcode'#1=13\relax
541 }
542 \@makeactive\,
543 \def,{\errmessage{COMMA}}
544 \@makeother\,
545 \@makeactive\=
546 \def={\errmessage{EQUALS}}
547 \@makeother\=
548
549 \begin{qstest}{normalize}{normalize,active-chars,space-removal}%
550     \def\Test#1#2{%
551         \@makeother\,%
552         \@makeother\=%
553         \scantokens{\toks@={#2}}%
554         \edef\Result{\the\toks@}%
555         \@makeother\,%
556         \@makeother\=%
557         \@Test{#1}%
558         \@makeactive\,%
559         \@Test{#1}%
560         \@makeactive\=%
561         \@Test{#1}%
562         \@makeother\,%
563         \@Test{#1}%
564         \@makeother\=%
565     }%
566     \def\@Test#1{%
567         \scantokens{\kv@normalize{#1}}%
568         \expandafter\expandafter\expandafter\Expect
569         \expandafter\expandafter\expandafter
570         {\expandafter\kv@list\expandafter}\expandafter{\Result}%
571         \Expect*{\ifx\kv@list\Result true\else false\fi}{true}%
572     }%
573     \Test{},{,%}
574     \Test{,},{,%}
575     \Test{,,},{,%}
576     \Test{,,},{,%}
577     \Test{ , }{,%}
578     \Test{{a}}{{a},{,%}
579     \Test{,{a}}{{a},{,%}
580     \Test{{a},{},{a},{,%}
581     \Test{{a},{b}}{{a},{b},{,%}
582     \Test{{b}={c},{},{d}={},{b}={c},{},{d}={},{d}={},{,%}
583     \Test{{}}{},{,%}
584     \Test{{},{},{},{},{},{},{,%}
585     \Test{=}{,=,%}
586     \Test{=,=}{,=,=,%}
587     \def\TestSet#1{%
588         \Test{#1#1}{,%}

```

```

589 \Test{#1#1,#1#1}{,}%
590 \Test{#1#1,#1#1,#1#1}{,}%
591 \Test{#1#1#1#1#1}{,}%
592 \Test{{a}#1#1=#1#1{b}}{,{a}={b}},}%
593 }%
594 \TestSet{ }%
595 \begingroup
596 \let\saved@normalize\kv@normalize
597 \def\kv@normalize#1{%
598   \saved@normalize{#1}%
599   \@onelevel@sanitize\kv@list
600   \@onelevel@sanitize\Result
601 }%
602 \Test{#,#=#,{#}={#},{#}=#,{#}}{, #,#=#,{#}={#},{#}=#,{#}},}%
603 \endgroup
604 \begingroup
605 \def\Test#1#2{%
606   \edef\Result{#2}%
607   \@Test{#1}%
608 }%
609 \Test{{ a = b }}{,{ a = b }},}%
610 \@makeactive\,%
611 \Test{{,}}{\string,{\noexpand,}\string,}%
612 \@makeother\,%
613 \@makeactive\=%
614 \Test{a={}}{,a\string={\noexpand=}},}%
615 \endgroup
616 \Test{a=b}{,a=b},}%
617 \Test{a={b}}{,a={b}},}%
618 \Test{a = {b}}{,a={b}},}%
619 \Test{a= {b}}{,a={b}},}%
620 \Test{a = {b}}{,a={b}},}%
621 \Test{a = {b} ,}{,a={b}},}%
622 \Test{a}{,a},}%
623 \Test{ a}{,a},}%
624 \Test{a }{,a},}%
625 \Test{ a }{,a},}%
626 \Test{, a ,}{,a},}%
627 \Test{, a b ,}{,a b},}%
628 \Test{,a ,}{,a},}%
629 \Test{ a =}{,a=},}%
630 \Test{ a = }{,a=},}%
631 \Test{a = }{,a=},}%
632 \Test{{a} = }{,{a}=},}%
633 \Test{{a}= {} }{,{a}={}},}%
634 \Test{, a = {} }{,a={}},}%
635 \Test{a,,b}{,a,b},}%
636 \Test{a=\fi}{,a=\fi},}%
637 \Test{a=\iffalse}{,a=\iffalse},}%
638 \Test{a=\iffalse,b=\fi}{,a=\iffalse,b=\fi},}%
639 \end{qstest}
640
641 \begin{qstest}{parse}{parse,brace-removal}
642 \def\Processor#1#2{%
643   \expandafter\Expect\expandafter{\kv@key}{#1}%
644   \toks@{#2}%
645   \edef\x{\the\toks@}%
646   \ifx\kv@value\relax
647     \Expect*{\the\toks@}{}%
648     \def\Value{<>}%
649   \else
650     \edef\Value{[\the\toks@]}%

```

```

651     \@onelevel@sanitize\Value
652 \fi
653 \toks@{#1}%
654 \ifx\Result\@empty
655     \edef\Result{[\the\toks@]=\Value}%
656 \else
657     \edef\Result{\Result, [\the\toks@]=\Value}%
658 \fi
659 \@onelevel@sanitize\Result
660 }%
661 \def\Test#1#2{%
662     \sbox0{%
663         \let\Result\@empty
664         \kv@parse{#1}\Processor
665         \Expect*{\Result}{#2}%
666     }%
667     \Expect*{\the\wd0}{0.0pt}%
668 }%
669 \Test{}{}%
670 \Test{{}}{}%
671 \Test{{{}}}{[]=<>}%
672 \Test{{{}}}{[{}]=<>}%
673 \Test{a}{[a]=<>}%
674 \Test{{a}}{[a]=<>}%
675 \Test{{{a}}}{[a]=<>}%
676 \Test{{{a}}}{[a]=<>}%
677 \Test{{{a}}}{[[a]=<>}%
678 \Test{a=}{[a]=[]}%
679 \Test{{a}=}{[a]=[]}%
680 \Test{{{a}}=}{[[a]=[]}%
681 \Test{a={}}{[a]=[]}%
682 \Test{{a}={}}{[a]=[{}]}%
683 \Test{a=b}{[a]=[b]}%
684 \Test{a=\fi}{[a]=[\fi]}%
685 \Test{a=\iffalse}{[a]=[\iffalse]}%
686 \Test{a=\iffalse,b=\fi}{[a]=[\iffalse],[b]=[\fi]}%
687 \Test{{ a = b }}{[ a ]=[ b ]}%
688 \Test{{{ a = b }}}{[ a = b ]=<>}%
689 \end{qstest}
690
691 \begin{qstest}{comma}{comma,parse}
692     \def\Processor#1{%
693         \expandafter\Expect\expandafter{\comma@entry}{#1}%
694         \toks@{#1}%
695         \ifx\Result\@empty
696             \edef\Result{[\the\toks@]}%
697         \else
698             \edef\Result{\Result, [\the\toks@]}%
699         \fi
700         \@onelevel@sanitize\Result
701     }%
702     \def\Test#1#2{%
703         \sbox0{%
704             \let\Result\@empty
705             \comma@parse{#1}\Processor
706             \Expect*{\Result}{#2}%
707         }%
708         \Expect*{\the\wd0}{0.0pt}%
709     }%
710     \Test{}{}%
711     \Test{{}}{}%
712     \Test{{{}}}{[{}]}%

```

```

713 \Test{a}{[a]}%
714 \Test{{a}}{[a]}%
715 \Test{{{a}}}{[a]}%
716 \Test{a=}{[a=]}%
717 \Test{a\fi}{[a\fi]}%
718 \Test{a\iffalse}{[a\iffalse]}%
719 \Test{\iffalse,\fi}{[\iffalse],[\fi]}%
720 \Test{ a , b , c }{[a],[b],[c]}%
721 \Test{ { } ,{ } , { } , { } , { } }{[ ],[ ],[ ],[ ],[ ]}%
722 \Test{ {{}} ,{{}} , {{}} , {{}} , {{}} }{[{}],[{}],[{}],[{}],[{}]}%
723 \end{qstest}
724
725 \begin{document}
726 \end{document}
727 </test2>

```

5 Installation

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

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

5.2 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/
```

5.3 Package installation

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

```
tex kvsetkeys.dtx
```

¹<http://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):

```
kvsetkeys.sty      → tex/generic/oberdiek/kvsetkeys.sty
kvsetkeys.pdf      → doc/latex/oberdiek/kvsetkeys.pdf
kvsetkeys-example.tex → doc/latex/oberdiek/kvsetkeys-example.tex
test/kvsetkeys-test1.tex → doc/latex/oberdiek/test/kvsetkeys-test1.tex
test/kvsetkeys-test2.tex → doc/latex/oberdiek/test/kvsetkeys-test2.tex
test/kvsetkeys-test3.tex → doc/latex/oberdiek/test/kvsetkeys-test3.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`.

5.4 Refresh file name databases

If your $\text{T}_{\text{E}}\text{X}$ distribution (`te $\text{T}_{\text{E}}\text{X}$` , `mik $\text{T}_{\text{E}}\text{X}$` , ...) relies on file name databases, you must refresh these. For example, `te $\text{T}_{\text{E}}\text{X}$` users run `texhash` or `mktextlsr`.

5.5 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 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$. The `.dtx` chooses its action depending on the format:
plain- $\text{T}_{\text{E}}\text{X}$: Run `docstrip` and extract the files.

$\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$: Generate the documentation.

If you insist on using $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ for `docstrip` (really, `docstrip` does not need $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$), 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 `pdf $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$` :

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

6 References

- [1] A guide to key-value methods, Joseph Wright, second draft for `TUG-Boat`, 2009-03-17. <http://www.texdev.net/wp-content/uploads/2009/03/keyval.pdf>
- [2] David Carlisle: *The keyval package*; 1999/03/16 v1.13; `CTAN:macros/latex/required/graphics/keyval.dtx`.

7 History

[2006/03/06 v1.0]

- First version.

[2006/10/19 v1.1]

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

[2007/09/09 v1.2]

- Using package `infwarerr` for error messages.
- Catcode section rewritten.

[2007/09/29 v1.3]

- Normalizing and parsing of comma separated lists added.
- `\kv@normalize` rewritten.
- Robustness increased for normalizing and parsing, e.g. for values with unmatched conditionals.
- ε -TeX is used if available.
- Tests added for normalizing and parsing.

[2009/07/19 v1.4]

- Bug fix for `\kv@normalize`: unwanted space removed (Florent Chervet).

[2009/07/30 v1.5]

- Documentation addition: recommendation for Joseph Wright's review article.

[2009/12/12 v1.6]

- Short info shortened.

[2009/12/22 v1.7]

- Internal optimization (`\KVS@CommaSpace`, ..., `\KVS@EqualsSpace`).

8 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>	412, 474	<code>\=</code> 36, 256, 257, 545, 547, 552, 556, 560, 564, 613
<code>\\$</code>	161, 164	<code>\></code> 38, 39, 40, 41, 42, 43
<code>\%</code>	477	<code>\@</code> 413, 470
<code>\&</code>	162, 165	<code>\@PackageError</code> 396
<code>\,</code>	208, 209, 542, 544, 551, 555, 558, 562, 610, 612	<code>\@Test</code> 557, 559, 561, 563, 566, 607
		<code>\@ehc</code> 401

293, 295, 300, 308, 310, 313, 317, 320, 326, 329, 343, 345, 353	\StopTime 518, 530
\KVS@Parse 308, <u>310</u>	\strip@pt 510
\KVS@Process 313, <u>320</u>	\SummaryTime 505, 507, 520, 534
\KVS@SecondOfTwo <u>169</u> , 179, 268	
\KVS@SetFamilyHandler 384, <u>386</u>	T
\KVS@SpaceComma 186, 200, <u>223</u>	\tag 8, 37, 39, 40, 42, 44
\KVS@SpaceEquals 190, <u>281</u>	\Test .. 465, 483, 550, 573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584, 585, 586, 588, 589, 590, 591, 592, 602, 605, 609, 611, 614, 616, 617, 618, 619, 620, 621, 622, 623, 624, 625, 626, 627, 628, 629, 630, 631, 632, 633, 634, 635, 636, 637, 638, 661, 669, 670, 671, 672, 673, 674, 675, 676, 677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 702, 710, 711, 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 722
\KVS@Temp 171, 174, 176, 264, 265	\TestSet 587, 594
\KVS@TestMode 494	\TestTime 506, 519, 520, 521
\kvsetkeys 4, 14, <u>404</u>	\textgreater 16
	\textless 16
L	\texttt 15
\lccode 208, 209, 256, 257	\the 16, 24, 115, 116, 117, 118, 129, 174, 192, 203, 213, 216, 224, 231, 235, 242, 246, 252, 261, 264, 273, 282, 289, 293, 300, 333, 448, 554, 645, 647, 650, 655, 657, 667, 696, 698, 708
\LoadCommand 463, 480	\TimeDescription 514, 517, 521
\LogTests 500	\TMP@EnsureCode 126, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142
\loop 429, 445, 456	\toks@ 12, 16, 23, 24, 173, 174, 184, 192, 198, 203, 212, 213, 216, 224, 228, 230, 231, 235, 239, 241, 242, 246, 249, 252, 260, 261, 264, 271, 273, 282, 286, 288, 289, 293, 297, 299, 300, 332, 333, 553, 554, 644, 645, 647, 650, 653, 655, 657, 694, 696, 698
\lowercase 210, 258	\typeout 509
	U
M	\UNDEFINED 492
\makeatletter 7, 493, 504, 538	\unexpanded 491, 492, 497
\makeatother 32, 495, 536	\unless 357, 360, 370
\mbox 36	\usepackage 3, 4, 5, 496, 498
\MessageBreak 399, 400	
	V
N	\Value 648, 650, 651, 655, 657
\NeedsTeXFormat 488	
\newcommand 8, 508, 513, 517, 518	W
\newcount 505, 506	\wd 667, 708
\next 435, 437, 439	\write 67, 93
\nofiles 489	
\number 510	X
	\x 57, 58, 61, 66, 70, 72, 92, 97, 107, 113, 121, 645
P	
\PackageInfo 70	Z
\pdfelapsedtime 519	\z@ 507
\pdfresettimer 515	
\PrintTime 508, 521, 534	
\Processor 642, 664, 692, 705	
\ProvidesPackage 62, 108	
Q	
\qqquad 36	
\qstest 523, 525	
R	
\RangeCatcodeInvalid 454, 466, 467, 468, 469	
\renewcommand 514	
\repeat 429, 441, 452, 460	
\RequirePackage 148, 149	
\RestoreCatcodes .. 443, 446, 447, 481	
\Result 554, 570, 571, 600, 606, 654, 655, 657, 659, 663, 665, 695, 696, 698, 700, 704, 706	
S	
\saved@endqstest 524, 531	
\saved@normalize 596, 598	
\saved@qstest 523, 526	
\SavedUnexpanded 491, 497	
\sbox 662, 703	
\scantokens 553, 567	
\space 25, 510	
\StartTime 513, 527	