

The kvsetkeys package

Heiko Oberdiek
<oberdiek@uni-freiburg.de>

2007/09/29 v1.3

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	2
1.3	Parsing key value lists	3
1.4	Processing key value pairs	3
1.5	Default family handler	4
1.6	Put it all together	4
1.7	Comma separated lists	4
2	Example	5
3	Implementation	5
3.1	Identification	5
3.2	Package loading	7
3.3	Check for ε -TeX	7
3.4	Generic help macros	8
3.5	Normalizing	8
3.6	Parsing key value lists	11
3.7	Parsing comma lists	11
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	15
5	Installation	19
5.1	Download	19
5.2	Bundle installation	19
5.3	Package installation	19
5.4	Refresh file name databases	19
5.5	Some details for the interested	20

6	References	20
7	History	20
	[2006/03/06 v1.0]	20
	[2006/10/19 v1.1]	20
	[2007/09/09 v1.2]	20
	[2007/09/29 v1.3]	21
8	Index	21

1 Documentation

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}}}
```

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  $\langle key \rangle$  exists
  call the keyval code of  $\langle key \rangle$ 
else
  if  $\langle handler \rangle$  for  $\langle family \rangle$  exists
     $\langle handler \rangle \{ \langle key \rangle \} \{ \langle value \rangle \}$ 
  else
    raise unknown key error
  fi
fi

```

1.5 Default family handler

`\kv@processor@default` calls $\langle handler \rangle$, 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 { $\langle family \rangle$ } { $\langle handler definition \rangle$ }
```

This sets the default family handler for the keyval family $\langle family \rangle$. Inside $\langle handler definition \rangle$ #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 { $\langle family \rangle$ } { $\langle key value list \rangle$ }
```

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

```
\kv@parse { $\langle key value list \rangle$ } { \kv@processor@default { $\langle family \rangle$ } }
```

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 \*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     \catcode35 6 % #
66     \catcode123 1 % {
67     \catcode125 2 % }
68     \expandafter\ifx\csname PackageInfo\endcsname\relax
69     \def\x#1#2{%
70       \immediate\write-1{Package #1 Info: #2.}%
71     }%
72   \else
73     \def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
74   \fi
75   \x{kvsetkeys}{The package is already loaded}%
76 \endgroup
77 \expandafter\endinput
78 \fi
79 \endgroup
```

Package identification:

```
80 \begingroup
81   \catcode35 6 % #
82   \catcode40 12 % (
83   \catcode41 12 % )
84   \catcode44 12 % ,
85   \catcode45 12 % -
86   \catcode46 12 % .
87   \catcode47 12 % /
88   \catcode58 12 % :
89   \catcode64 11 % @
90   \catcode123 1 % {
91   \catcode125 2 % }
92   \expandafter\ifx\csname ProvidesPackage\endcsname\relax
93     \def\x#1#2#3[#4]{\endgroup
94       \immediate\write-1{Package: #3 #4}%
95       \xdef#1{#4}%
96     }%
97   \else
98     \def\x#1#2[#3]{\endgroup
99       #2[#3]}%
100     \ifx#1\@undefined
101       \xdef#1{#3}%
102     \fi
103     \ifx#1\relax
104       \xdef#1{#3}%
105     \fi
106   }%
107 \fi
```

```

108 \expandafter\x\csname ver@kvsetkeys.sty\endcsname
109 \ProvidesPackage{kvsetkeys}%
110 [2007/09/29 v1.3 Key value parser with default handler support (H0)]
111 \begingroup
112 \catcode123 1 % {
113 \catcode125 2 % }
114 \def\x{\endgroup
115 \expandafter\edef\csname KVS@AtEnd\endcsname{%
116 \catcode35 \the\catcode35\relax
117 \catcode64 \the\catcode64\relax
118 \catcode123 \the\catcode123\relax
119 \catcode125 \the\catcode125\relax
120 }%
121 }%
122 \x
123 \catcode35 6 % #
124 \catcode64 11 % @
125 \catcode123 1 % {
126 \catcode125 2 % }
127 \def\TMP@EnsureCode#1#2{%
128 \edef\KVS@AtEnd{%
129 \KVS@AtEnd
130 \catcode#1 \the\catcode#1\relax
131 }%
132 \catcode#1 #2\relax
133 }
134 \TMP@EnsureCode{36}{3}% $
135 \TMP@EnsureCode{38}{4}% &
136 \TMP@EnsureCode{39}{12}% '
137 \TMP@EnsureCode{44}{12}% ,
138 \TMP@EnsureCode{46}{12}% .
139 \TMP@EnsureCode{47}{12}% /
140 \TMP@EnsureCode{61}{12}% =
141 \TMP@EnsureCode{94}{7}% ^ (superscript)
142 \TMP@EnsureCode{96}{12}% '
143 \TMP@EnsureCode{126}{13}% ~ (active)

```

3.2 Package loading

```

144 \begingroup\expandafter\expandafter\expandafter\endgroup
145 \expandafter\ifx\csname RequirePackage\endcsname\relax
146 \input infwarerr.sty\relax
147 \input etexcmds.sty\relax
148 \else
149 \RequirePackage{infwarerr}[2007/09/09]%
150 \RequirePackage{etexcmds}[2007/09/09]%
151 \fi

```

3.3 Check for ϵ -TeX

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

```

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

```

```

165 \catcode'\$=14 % comment
166 \catcode'\&=9 % ignore
167 \fi

```

3.4 Generic help macros

```

\KVS@Empty
168 \def\KVS@Empty{}

\KVS@FirstOfTwo
169 \long\def\KVS@FirstOfTwo#1#2{#1}

\KVS@SecondOfTwo
170 \long\def\KVS@SecondOfTwo#1#2{#2}

\KVS@IfEmpty
171 \def\KVS@IfEmpty#1{%
172 & \edef\KVS@Temp{\etex@unexpanded{#1}}%
173 $ \begingroup
174 $ \toks@{#1}%
175 $ \edef\KVS@Temp{\the\toks@}%
176 $ \expandafter\endgroup
177 \ifx\KVS@Temp\KVS@Empty
178 \expandafter\KVS@FirstOfTwo
179 \else
180 \expandafter\KVS@SecondOfTwo
181 \fi
182 }

```

3.5 Normalizing

```

\kv@normalize
183 \def\kv@normalize#1{%
184 \begingroup
185 \toks@{, #1,}%
186 \KVS@Comma
187 \KVS@SpaceComma{ }%
188 \KVS@CommaSpace
189 \KVS@CommaComma
190 \KVS@Equals
191 \KVS@SpaceEquals{ }%
192 \KVS@EqualsSpace{ }%
193 \xdef\KVS@Global{\the\toks@}%
194 \endgroup
195 \let\kv@list\KVS@Global
196 }

\comma@normalize
197 \def\comma@normalize#1{%
198 \begingroup
199 \toks@{, #1,}%
200 \KVS@Comma
201 \KVS@SpaceComma{ }%
202 \KVS@CommaSpace
203 \KVS@CommaComma
204 \xdef\KVS@Global{\the\toks@}%
205 \endgroup
206 \let\comma@list\KVS@Global
207 }

```

`\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.

```

208 \begingroup
209   \lccode'\,='\",%
210   \lccode'\~='\",%
211 \lowercase{\endgroup
212   \def\KVS@Comma{%
213     \toks@\expandafter{\expandafter}\expandafter
214     \KVS@@Comma\the\toks@\KVS@Nil
215   }%
216   \def\KVS@@Comma#1~#2\KVS@Nil{%
217     \toks@\expandafter{\the\toks@#1}%
218     \KVS@IfEmpty{#2}{%
219       }{%
220         \KVS@@Comma,#2\KVS@Nil
221       }%
222     }%
223 }

```

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

```

224 \def\KVS@SpaceComma#1{%
225   \toks@\expandafter{\the\toks@#1,}%
226   \expandafter\KVS@@SpaceComma\the\toks@\KVS@Nil
227 }

```

`\KVS@@SpaceComma`

```

228 \def\KVS@@SpaceComma#1 ,#2\KVS@Nil{%
229   \KVS@IfEmpty{#2}{%
230     \toks@{#1}%
231   }{%
232     \toks@{#1,#2}%
233     \expandafter\KVS@@SpaceComma\the\toks@\KVS@Nil
234   }%
235 }

```

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

```

236 \def\KVS@CommaSpace{%
237   \toks@\expandafter{\the\toks@,}%
238   \expandafter\KVS@@CommaSpace\the\toks@\KVS@Nil
239 }

```

`\KVS@@CommaSpace`

```

240 \def\KVS@@CommaSpace#1 ,#2\KVS@Nil{%
241   \KVS@IfEmpty{#2}{%
242     \toks@{#1}%
243   }{%
244     \toks@{#1,#2}%
245     \expandafter\KVS@@CommaSpace\the\toks@\KVS@Nil
246   }%
247 }

```

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

```

248 \def\KVS@CommaComma{%
249   \toks@\expandafter{\the\toks@,}%
250   \expandafter\KVS@@CommaComma\the\toks@\KVS@Nil
251 }

```

`\KVS@@CommaComma`

```

252 \def\KVS@@CommaComma#1,,#2\KVS@Nil{%
253   \toks@{#1,#2}%
254   \KVS@IfEmpty{#2}{%

```

```

255 }{%
256   \expandafter\KVS@@CommaComma\the\toks@\KVS@Nil
257 }%
258 }

```

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

```

259 \begingroup
260   \lccode'\=='\=%
261   \lccode'\~='\=%
262 \lowercase{\endgroup
263   \def\KVS@Equals{%
264     \toks@\expandafter{\expandafter}\expandafter
265     \KVS@@Equals\the\toks@~\KVS@Nil
266   }%
267   \def\KVS@@Equals#1~#2\KVS@Nil{%
268     \edef\KVS@Temp{\the\toks@}%
269     \ifx\KVS@Temp\KVS@Empty
270       \expandafter\KVS@FirstOfTwo
271     \else
272       \expandafter\KVS@SecondOfTwo
273     \fi
274     {%
275       \toks@{#1}%
276     }{%
277       \toks@\expandafter{\the\toks@=#1}%
278     }%
279     \KVS@IfEmpty{#2}{%
280     }{%
281       \KVS@@Equals#2\KVS@Nil
282     }%
283   }%
284 }

```

\KVS@SpaceEquals Removes spaces before the equals sign.

```

285 \def\KVS@SpaceEquals#1{%
286   \toks@\expandafter{\the\toks@#1}%
287   \expandafter\KVS@@SpaceEquals\the\toks@\KVS@Nil
288 }

```

\KVS@@SpaceEquals

```

289 \def\KVS@@SpaceEquals#1=#2\KVS@Nil{%
290   \KVS@IfEmpty{#2}{%
291     \toks@{#1}%
292   }{%
293     \toks@{#1=#2}%
294     \expandafter\KVS@@SpaceEquals\the\toks@\KVS@Nil
295   }%
296 }

```

\KVS@EqualsSpace Removes spaces after the equals sign.

```

297 \def\KVS@EqualsSpace{%
298   \toks@\expandafter{\the\toks@= }%
299   \expandafter\KVS@@EqualsSpace\the\toks@\KVS@Nil
300 }

```

\KVS@@EqualsSpace

```

301 \def\KVS@@EqualsSpace#1=#2\KVS@Nil{%
302   \KVS@IfEmpty{#2}{%
303     \toks@{#1}%
304   }{%
305     \toks@{#1=#2}%

```

```

306     \expandafter\KVS@@EqualsSpace\the\toks@\KVS@Nil
307 }%
308 }

```

3.6 Parsing key value lists

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

```

309 \def\kv@parse#1{%
310   \kv@normalize{#1}%
311   \expandafter\kv@parse@normalized\expandafter{\kv@list}%
312 }

```

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

```

313 \def\kv@parse@normalized#1#2{%
314   \KVS@Parse#1,\KVS@Nil{#2}%
315 }

```

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

```

316 \def\KVS@Parse#1,#2\KVS@Nil#3{%
317   \KVS@IfEmpty{#1}{%
318     }{%
319     \KVS@Process#1=\KVS@Nil{#3}%
320   }%
321   \KVS@IfEmpty{#2}{%
322     }{%
323     \KVS@Parse#2\KVS@Nil{#3}%
324   }%
325 }

```

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

```

326 \def\KVS@Process#1=#2\KVS@Nil#3{%
327   \def\kv@key{#1}%
328   \KVS@IfEmpty{#2}{%
329     \let\kv@value\relax
330     #3{#1}{}%
331   }{%
332     \KVS@@Process{#1}#2\KVS@Nil{#3}%
333   }%
334 }

```

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

```

335 \def\KVS@@Process#1#2=\KVS@Nil#3{%
336   & \edef\kv@value{\etex@unexpanded{#2}}%
337   $ \begingroup
338   $ \toks@{#2}%
339   $ \xdef\KVS@Global{\the\toks@}%
340   $ \endgroup
341   $ \let\kv@value\KVS@Global
342   #3{#1}{#2}%
343 }

```

3.7 Parsing comma lists

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

```

344 \def\comma@parse#1{%
345   \comma@normalize{#1}%
346   \expandafter\comma@parse@normalized\expandafter{\comma@list}%
347 }

\comma@parse@normalized #1: comma list
#2: processor

348 \def\comma@parse@normalized#1#2{%
349   \KVS@CommaParse#1,\KVS@Nil{#2}%
350 }

\KVS@CommaParse #1,#2: comma list
#3: processor

351 \def\KVS@CommaParse#1,#2\KVS@Nil#3{%
352   \KVS@IfEmpty{#1}{%
353   }{%
354     \def\comma@entry{#1}%
355     #3{#1}%
356   }%
357   \KVS@IfEmpty{#2}{%
358   }{%
359     \KVS@CommaParse#2\KVS@Nil{#3}%
360   }%
361 }

```

3.8 Processing key value pairs

\kv@processor@default

```

362 \def\kv@processor@default#1#2#3{%
363   & \unless\ifcsname KV@#1@#2\endcsname
364   $ \begingroup\expandafter\expandafter\expandafter\endgroup
365   $ \expandafter\ifx\csname KV@#1@#2\endcsname\relax
366   & \unless\ifcsname KVS@#1@handler\endcsname
367   $ \begingroup\expandafter\expandafter\expandafter\endgroup
368   $ \expandafter\ifx\csname KVS@#1@handler\endcsname\relax
369     \kv@error@unknownkey{#1}{#2}%
370   \else
371     \csname KVS@#1@handler\endcsname{#2}{#3}%
372     \relax
373   \fi
374   \else
375     \ifx\kv@value\relax
376     & \unless\ifcsname KV@#1@#2@default\endcsname
377     $ \begingroup\expandafter\expandafter\expandafter\endgroup
378     $ \expandafter\ifx\csname KV@#1@#2@default\endcsname\relax
379       \kv@error@novalue{#1}{#2}%
380     \else
381       \csname KV@#1@#2@default\endcsname
382       \relax
383     \fi
384     \else
385       \csname KV@#1@#2\endcsname{#3}%
386     \fi
387   \fi
388 }

```

\kv@set@family@handler

```

389 \def\kv@set@family@handler#1{%
390   \KVS@SetFamilyHandler{#1}\@nil
391 }

```

\KVS@SetFamilyHandler

```
392 \def\KVS@SetFamilyHandler#1\@nil#{%
393   \expandafter\def\csname KVS@#1@handler\endcsname##1##2%
394 }
```

3.9 Error handling

\kv@error@novalue

```
395 \def\kv@error@novalue{%
396   \kv@error@generic{No value specified for}%
397 }
```

\kv@error@unknownkey

```
398 \def\kv@error@unknownkey{%
399   \kv@error@generic{Undefined}%
400 }
```

\kv@error@generic

```
401 \def\kv@error@generic#1#2#3{%
402   \@PackageError{kvsetkeys}{%
403     #1 key ‘#3’%
404   }{%
405     The keyval family of the key ‘#3’ is ‘#2’.\MessageBreak
406     \MessageBreak
407     \@ehc
408   }%
409 }
```

3.10 Do it all

\kvsetkeys

```
410 \def\kvsetkeys#1#2{%
411   \kv@parse{#2}{\kv@processor@default{#1}}%
412 }

413 \KVS@AtEnd
414 \</package>
```

4 Test

4.1 Catcode checks for loading

```
415 <*test1>

416 \catcode‘\{=1 %
417 \catcode‘\}=2 %
418 \catcode‘\#=6 %
419 \catcode‘\@=11 %
420 \expandafter\ifx\csname count@\endcsname\relax
421   \countdef\count@=255 %
422 \fi
423 \expandafter\ifx\csname @gobble\endcsname\relax
424   \long\def@gobble#1{%
425 \fi
426 \expandafter\ifx\csname @firstofone\endcsname\relax
427   \long\def@firstofone#1{#1}%
428 \fi
429 \expandafter\ifx\csname loop\endcsname\relax
430   \expandafter@firstofone
431 \else
```

```

432 \expandafter\@gobble
433 \fi
434 {%
435 \def\loop#1\repeat{%
436 \def\body{#1}%
437 \iterate
438 }%
439 \def\iterate{%
440 \body
441 \let\next\iterate
442 \else
443 \let\next\relax
444 \fi
445 \next
446 }%
447 \let\repeat=\fi
448 }%
449 \def\RestoreCatcodes{}
450 \count@=0 %
451 \loop
452 \edef\RestoreCatcodes{%
453 \RestoreCatcodes
454 \catcode\the\count@=\the\catcode\count@\relax
455 }%
456 \ifnum\count@<255 %
457 \advance\count@ 1 %
458 \repeat
459
460 \def\RangeCatcodeInvalid#1#2{%
461 \count@=#1\relax
462 \loop
463 \catcode\count@=15 %
464 \ifnum\count@<#2\relax
465 \advance\count@ 1 %
466 \repeat
467 }
468 \expandafter\ifx\csname LoadCommand\endcsname\relax
469 \def\LoadCommand{\input kvsetkeys.sty\relax}%
470 \fi
471 \def\Test{%
472 \RangeCatcodeInvalid{0}{47}%
473 \RangeCatcodeInvalid{58}{64}%
474 \RangeCatcodeInvalid{91}{96}%
475 \RangeCatcodeInvalid{123}{255}%
476 \catcode'\@=12 %
477 \catcode'\=0 %
478 \catcode'\{=1 %
479 \catcode'\}=2 %
480 \catcode'\#=6 %
481 \catcode'\[=12 %
482 \catcode'\]=12 %
483 \catcode'\%=14 %
484 \catcode'\ =10 %
485 \catcode13=5 %
486 \LoadCommand
487 \RestoreCatcodes
488 }
489 \Test
490 \csname @@end\endcsname
491 \end
492 </test1>

```

4.2 Macro tests

4.2.1 Preamble

```
493 <*test2>
494 \NeedsTeXFormat{LaTeX2e}
495 \nofiles
496 \documentclass{article}
497 <noetex>\let\SavedUnexpanded\unexpanded
498 <noetex>\let\unexpanded\UNDEFINED
499 \makeatletter
500 \chardef\KVS@TestMode=1 %
501 \makeatother
502 \usepackage{kvsetkeys}[2007/09/29]
503 <noetex>\let\unexpanded\SavedUnexpanded
504 \usepackage{qstest}
505 \IncludeTests{*}
506 \LogTests{log}{*}{*}
```

4.2.2 Time

```
507 \begingroup\expandafter\expandafter\expandafter\endgroup
508 \expandafter\ifx\csname pdfresettimer\endcsname\relax
509 \else
510   \makeatletter
511   \newcount\SummaryTime
512   \newcount\TestTime
513   \SummaryTime=\z@
514   \newcommand*\PrintTime[2]{%
515     \typeout{%
516       [Time #1: \strip@pt\dimexpr\number#2sp\relax\space s]%
517     }%
518   }%
519   \newcommand*\StartTime[1]{%
520     \renewcommand*\TimeDescription{#1}%
521     \pdfresettimer
522   }%
523   \newcommand*\TimeDescription{}%
524   \newcommand*\StopTime{%
525     \TestTime=\pdfelapsedtime
526     \global\advance\SummaryTime\TestTime
527     \PrintTime\TimeDescription\TestTime
528   }%
529   \let\saved@qstest\qstest
530   \let\saved@endqstest\endqstest
531   \def\qstest#1#2{%
532     \saved@qstest{#1}{#2}%
533     \StartTime{#1}%
534   }%
535   \def\endqstest{%
536     \StopTime
537     \saved@endqstest
538   }%
539   \AtEndDocument{%
540     \PrintTime{summary}\SummaryTime
541   }%
542   \makeatother
543 \fi
```

4.2.3 Test sets

```
544 \makeatletter
545 \def\@makeactive#1{%
546   \catcode'#1=13\relax
547 }
```

```

548 \@makeactive\,
549 \def,{\errmessage{COMMA}}
550 \@makeother\,
551 \@makeactive\=
552 \def={\errmessage{EQUALS}}
553 \@makeother\=
554
555 \begin{qstest}{normalize}{normalize,active-chars,space-removal}%
556   \def\Test#1#2{%
557     \@makeother\,%
558     \@makeother\=%
559     \scantokens{\toks@={#2}}%
560     \edef\Result{\the\toks@}%
561     \@makeother\,%
562     \@makeother\=%
563     \@Test{#1}%
564     \@makeactive\,%
565     \@Test{#1}%
566     \@makeactive\=%
567     \@Test{#1}%
568     \@makeother\,%
569     \@Test{#1}%
570     \@makeother\=%
571   }%
572   \def\@Test#1{%
573     \scantokens{\kv@normalize{#1}}%
574     \expandafter\expandafter\expandafter\Expect
575     \expandafter\expandafter\expandafter
576     {\expandafter\kv@list\expandafter}\expandafter{\Result}%
577     \Expect*{\ifx\kv@list\Result true\else false\fi}{true}%
578   }%
579   \Test{}{,}%
580   \Test{,}{,}%
581   \Test{,,}{,}%
582   \Test{,,}{,}%
583   \Test{ , }{,}%
584   \Test{{a}}{,{a},}%
585   \Test{,{a}}{,{a},}%
586   \Test{{a},}{,{a},}%
587   \Test{{a},{b}}{,{a},{b},}%
588   \Test{{b}={c},{},{}={},{d}={},{b}={c},{},{}={},{d}={},}%
589   \Test{{}}{,{},}%
590   \Test{{},{},{}}{,{},{},}%
591   \Test{=}{,=,}%
592   \Test{=,=}{,=,=,}%
593   \def\TestSet#1{%
594     \Test{#1#1}{,}%
595     \Test{#1#1,#1#1}{,}%
596     \Test{#1#1,#1#1,#1#1}{,}%
597     \Test{#1#1#1#1#1}{,}%
598     \Test{{a}#1#1=#1#1{b}}{,{a}={b},}%
599   }%
600   \TestSet{ }%
601   \begingroup
602     \let\saved@normalize\kv@normalize
603     \def\kv@normalize#1{%
604       \saved@normalize{#1}%
605       \@onelevel@sanitize\kv@list
606       \@onelevel@sanitize\Result
607     }%
608     \Test{#,#=#,{#}={#},{#}=#,{#}}{,###{#}={#},{#}=#,{#},}%
609   \endgroup

```

```

610 \begingroup
611   \def\Test#1#2{%
612     \edef\Result{#2}%
613     \@Test{#1}%
614   }%
615   \Test{{ a = b }}{,{ a = b },}%
616   \@makeactive\,%
617   \Test{{,}}{\string,{\noexpand,}\string,}%
618   \@makeother\,%
619   \@makeactive\=%
620   \Test{a={}}{,a\string={\noexpand=},}%
621 \endgroup
622 \Test{a=b}{,a=b,}%
623 \Test{a={b}}{,a={b},}%
624 \Test{a = {b}}{,a={b},}%
625 \Test{a= {b}}{,a={b},}%
626 \Test{a = {b}}{,a={b},}%
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 b ,}{,a b,}%
634 \Test{,a ,}{,a,}%
635 \Test{ a =}{,a=,}%
636 \Test{ a = }{,a=,}%
637 \Test{a =}{,a=,}%
638 \Test{{a} =}{,{a}=,}%
639 \Test{{a}= {}}{,{a}={},}%
640 \Test{, a = {}}{,a={},}%
641 \Test{a,,b}{,a,b,}%
642 \Test{a=\fi}{,a=\fi,}%
643 \Test{a=\iffalse}{,a=\iffalse,}%
644 \Test{a=\iffalse,b=\fi}{,a=\iffalse,b=\fi,}%
645 \end{qstest}
646
647 \begin{qstest}{parse}{parse,brace-removal}
648   \def\Processor#1#2{%
649     \expandafter\Expect\expandafter{\kv@key}{#1}%
650     \toks@{#2}%
651     \edef\x{\the\toks@}%
652     \ifx\kv@value\relax
653       \Expect*{\the\toks@}{}%
654       \def\Value{<>}%
655     \else
656       \edef\Value{[\the\toks@]}%
657       \@onelevel@sanitize\Value
658     \fi
659     \toks@{#1}%
660     \ifx\Result\@empty
661       \edef\Result{[\the\toks@]=\Value}%
662     \else
663       \edef\Result{\Result, [\the\toks@]=\Value}%
664     \fi
665     \@onelevel@sanitize\Result
666   }%
667   \def\Test#1#2{%
668     \let\Result\@empty
669     \kv@parse{#1}\Processor
670     \Expect*{\Result}{#2}%
671   }%

```

```

672 \Test{}{}%
673 \Test{{}}{}%
674 \Test{{{}}}{[]=<>}%
675 \Test{{{}}}{[{}]=<>}%
676 \Test{a}{[a]=<>}%
677 \Test{{a}}{[a]=<>}%
678 \Test{{{a}}}{[a]=<>}%
679 \Test{{{a}}}{[a]=<>}%
680 \Test{{{a}}}{[{}]=<>}%
681 \Test{a=}{[a]=[]}%
682 \Test{{a=}}{[a]=[]}%
683 \Test{{{a=}}}{[{}]=[]}%
684 \Test{a={}}{[a]=[]}%
685 \Test{{a}={}}{[a]=[{}]}%
686 \Test{a=b}{[a]=[b]}%
687 \Test{a=\fi}{[a]=[\fi]}%
688 \Test{a=\iffalse}{[a]=[\iffalse]}%
689 \Test{a=\iffalse,b=\fi}{[a]=[\iffalse],[b]=[\fi]}%
690 \Test{{ a = b }}{[ a ]=[ b ]}%
691 \Test{{{ a = b }}}{[ a = b ]=<>}%
692 \end{qstest}
693
694 \begin{qstest}{comma}{comma,parse}
695 \def\Processor#1{%
696 \expandafter\Expect\expandafter{\comma@entry}{#1}%
697 \toks@{#1}%
698 \ifx\Result\@empty
699 \edef\Result{[\the\toks@]}%
700 \else
701 \edef\Result{\Result,[\the\toks@]}%
702 \fi
703 \@onelevel@sanitize\Result
704 }%
705 \def\Test#1#2{%
706 \let\Result\@empty
707 \comma@parse{#1}\Processor
708 \Expect*{\Result}{#2}%
709 }%
710 \tracingmacros=1
711 \Test{}{}%
712 \Test{{}}{}%
713 \Test{{{}}}{[{}]}%
714 \Test{a}{[a]}%
715 \Test{{a}}{[a]}%
716 \Test{{{a}}}{[{}]}%
717 \Test{a=}{[a=]}%
718 \Test{a\fi}{[a\fi]}%
719 \Test{a\iffalse}{[a\iffalse]}%
720 \Test{\iffalse,\fi}{[\iffalse],[fi]}%
721 \Test{ a , b , c }{[a],[b],[c]}%
722 \Test{ { } , { } , { } , { } }{[ ],[ ],[ ],[ ],[ ]}%
723 \Test{ {{ } },{{ } }, {{ } }, {{ } } , {{ } } }{[{}],[{}],[{}],[{}],[{}]}%
724 \end{qstest}
725
726 \begin{document}
727 \end{document}
728 </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
```

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>test/kvsetkeys-test1.tex</code>	→ <code>doc/latex/oberdiek/test/kvsetkeys-test1.tex</code>
<code>test/kvsetkeys-test2.tex</code>	→ <code>doc/latex/oberdiek/test/kvsetkeys-test2.tex</code>
<code>test/kvsetkeys-test3.tex</code>	→ <code>doc/latex/oberdiek/test/kvsetkeys-test3.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`.

5.4 Refresh file name databases

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

¹<http://ftp.ctan.org/tex-archive/>

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 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
```

6 References

- [1] 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.

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	C
<code>\#</code> 418, 480	<code>\catcode</code> 50, 51, 52, 53, 54, 65, 66, 67, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 112, 113, 116, 117, 118, 119, 123, 124, 125, 126, 130, 132, 162, 163, 165, 166, 416, 417, 418, 419, 454, 463, 476, 477, 478, 479, 480, 481, 482, 483, 484, 485, 546
<code>\\$</code> 162, 165	<code>\chardef</code> 500
<code>\%</code> 483	<code>\comma@entry</code> 354, 696
<code>\&</code> 163, 166	<code>\comma@list</code> 206, 346
<code>\,</code> 209, 210, 548, 550, 557, 561, 564, 568, 616, 618	<code>\comma@normalize</code> 4, 197, 345
<code>\=</code> 36, 260, 261, 551, 553, 558, 562, 566, 570, 619	<code>\comma@parse</code> 4, 344, 707
<code>\></code> 38, 39, 40, 41, 42, 43	<code>\comma@parse@normalized</code> . 4, 346, 348
<code>\@</code> 419, 476	<code>\count@</code> 421, 450, 454, 456, 457, 461, 463, 464, 465
<code>\@PackageError</code> 402	<code>\countdef</code> 421
<code>\@Test</code> 563, 565, 567, 569, 572, 613	<code>\csname</code> 55, 68, 92, 108, 115, 145, 154, 156, 365, 368, 371, 378, 381, 385, 393, 420, 423, 426, 429, 468, 490, 508
<code>\@ehc</code> 407	
<code>\@empty</code> 13, 660, 668, 698, 706	D
<code>\@endslash</code> 13, 16, 30	<code>\define@key</code> 29
<code>\@firstofone</code> 427, 430	<code>\dimexpr</code> 516
<code>\@gobble</code> 424, 432	<code>\documentclass</code> 2, 496
<code>\@makeactive</code> 545, 548, 551, 564, 566, 616, 619	<code>\dots</code> 38, 41, 43
<code>\@makeother</code> 550, 553, 557, 558, 561, 562, 568, 570, 618	
<code>\@nil</code> 390, 392	E
<code>\@onelevel@sanitize</code> 605, 606, 657, 665, 703	<code>\empty</code> 59
<code>\@undefined</code> 100	<code>\end</code> 45, 46, 491, 645, 692, 724, 727
<code>\[</code> 481	<code>\endcsname</code> 55, 68, 92, 108, 115, 145, 154, 156, 363, 365, 366, 368, 371, 376, 378, 381, 385, 393, 420, 423, 426, 429, 468, 490, 508
<code>\]</code> 37, 38, 39, 40, 41, 42, 43, 44, 477	<code>\endinput</code> 77
<code>\{</code> 416, 478	<code>\endqstest</code> 530, 535
<code>\}</code> 417, 479	<code>\errmessage</code> 549, 552
<code>\]</code> 482	<code>\etex@unexpanded</code> 172, 336
<code>\~</code> 210, 261	<code>\Expect</code> 574, 577, 649, 653, 670, 696, 708
<code>_</code> 484	
	I
A	<code>\ifcase</code> 56, 153
<code>\advance</code> 457, 465, 526	
<code>\AtEndDocument</code> 539	
B	
<code>\begin</code> 34, 35, 555, 647, 694, 726	
<code>\body</code> 436, 440	

\ifcsname	363, 366, 376		
\ifetex@unexpanded	153		
\iffalse	643, 644, 688, 689, 719, 720		
\ifnum	456, 464		
\ifx	57, 59, 68, 92, 100, 103, 145, 154, 156, 177, 269, 365, 368, 375, 378, 420, 423, 426, 429, 468, 508, 577, 652, 660, 698		
\immediate	70, 94		
\IncludeTests	505		
\input	146, 147, 469		
\iterate	437, 439, 441		
K			
\kill	36		
\kv@error@generic	396, 399, 401		
\kv@error@novalue	379, 395		
\kv@error@unknownkey	369, 398		
\kv@key	327, 649		
\kv@list	195, 311, 576, 577, 605		
\kv@normalize	2, 183, 310, 573, 602, 603		
\kv@parse	3, 309, 411, 669		
\kv@parse@normalized	3, 311, 313		
\kv@processor@default	3, 362, 411		
\kv@set@family@handler	4, 20, 389		
\kv@value	329, 336, 341, 375, 652		
\KVS@@Comma	214, 216, 220		
\KVS@@CommaComma	250, 252		
\KVS@@CommaSpace	238, 240		
\KVS@@Equals	265, 267, 281		
\KVS@@EqualsSpace	299, 301		
\KVS@@Process	332, 335		
\KVS@@SpaceComma	226, 228		
\KVS@@SpaceEquals	287, 289		
\KVS@AtEnd	128, 129, 413		
\KVS@Comma	186, 200, 208		
\KVS@CommaComma	189, 203, 248		
\KVS@CommaParse	349, 351		
\KVS@CommaSpace	188, 202, 236		
\KVS@Empty	168, 177, 269		
\KVS@Equals	190, 259		
\KVS@EqualsSpace	192, 297		
\KVS@FirstOfTwo	169, 178, 270		
\KVS@Global	193, 195, 204, 206, 339, 341		
\KVS@IfEmpty	171, 218, 229, 241, 254, 279, 290, 302, 317, 321, 328, 352, 357		
\KVS@Nil	214, 216, 220, 226, 228, 233, 238, 240, 245, 250, 252, 256, 265, 267, 281, 287, 289, 294, 299, 301, 306, 314, 316, 319, 323, 326, 332, 335, 349, 351, 359		
\KVS@Parse	314, 316		
\KVS@Process	319, 326		
\KVS@SecondOfTwo	170, 180, 272		
\KVS@SetFamilyHandler	390, 392		
\KVS@SpaceComma	187, 201, 224		
\KVS@SpaceEquals	191, 285		
\KVS@Temp	172, 175, 177, 268, 269		
\KVS@TestMode	500		
\kvsetkeys	4, 14, 410		
L			
\lccode	209, 210, 260, 261		
\LoadCommand	469, 486		
\LogTests	506		
\loop	435, 451, 462		
\lowercase	211, 262		
M			
\makeatletter	7, 499, 510, 544		
\makeatother	32, 501, 542		
\mbox	36		
\MessageBreak	405, 406		
N			
\NeedsTeXFormat	494		
\newcommand	8, 514, 519, 523, 524		
\newcount	511, 512		
\next	441, 443, 445		
\nofiles	495		
\number	516		
P			
\PackageInfo	73		
\pdfelapsedtime	525		
\pdfresettimer	521		
\PrintTime	514, 527, 540		
\Processor	648, 669, 695, 707		
\ProvidesPackage	109		
Q			
\qqquad	36		
\qstest	529, 531		
R			
\RangeCatcodeInvalid	460, 472, 473, 474, 475		
\renewcommand	520		
\repeat	435, 447, 458, 466		
\RequirePackage	149, 150		
\RestoreCatcodes	449, 452, 453, 487		
\Result	560, 576, 577, 606, 612, 660, 661, 663, 665, 668, 670, 698, 699, 701, 703, 706, 708		
S			
\saved@endqstest	530, 537		
\saved@normalize	602, 604		
\saved@qstest	529, 532		
\SavedUnexpanded	497, 503		
\scantokens	559, 573		
\space	25, 516		
\StartTime	519, 533		
\StopTime	524, 536		
\strip@pt	516		
\SummaryTime	511, 513, 526, 540		
T			
\tag	8, 37, 39, 40, 42, 44		
\Test	471, 489, 556, 579, 580, 581, 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 594, 595, 596, 597, 598, 608, 611, 615, 617, 620, 622, 623, 624,		

625, 626, 627, 628, 629, 630,	238, 242, 244, 245, 249, 250,
631, 632, 633, 634, 635, 636,	253, 256, 264, 265, 268, 275,
637, 638, 639, 640, 641, 642,	277, 286, 287, 291, 293, 294,
643, 644, 667, 672, 673, 674,	298, 299, 303, 305, 306, 338,
675, 676, 677, 678, 679, 680,	339, 559, 560, 650, 651, 653,
681, 682, 683, 684, 685, 686,	656, 659, 661, 663, 697, 699, 701
687, 688, 689, 690, 691, 705,	\tracingmacros 710
711, 712, 713, 714, 715, 716,	\typeout 515
717, 718, 719, 720, 721, 722, 723	
\TestSet 593, 600	U
\TestTime 512, 525, 526, 527	\UNDEFINED 498
\textgreater 16	\unexpanded 497, 498, 503
\textless 16	\unless 363, 366, 376
\texttt 15	\usepackage 3, 4, 5, 502, 504
\the 16, 24, 116,	
117, 118, 119, 130, 175, 193,	V
204, 214, 217, 225, 226, 233,	\Value 654, 656, 657, 661, 663
237, 238, 245, 249, 250, 256,	
265, 268, 277, 286, 287, 294,	W
298, 299, 306, 339, 454, 560,	\write 70, 94
651, 653, 656, 661, 663, 699, 701	
\TimeDescription 520, 523, 527	X
\TMP@EnsureCode 127, 134, 135, 136,	\x 55, 57, 59,
137, 138, 139, 140, 141, 142, 143	69, 73, 75, 93, 98, 108, 114, 122, 651
\toks@ . 12, 16, 23, 24, 174, 175, 185,	
193, 199, 204, 213, 214, 217,	Z
225, 226, 230, 232, 233, 237,	\z@ 513