The \texttt{kvoptions} package

Heiko Oberdiek\textsuperscript{*}

2019/11/29 v3.13

Abstract

This package is intended for package authors who want to use options in key value format for their package options.

Contents

1 Introduction 3

1.1 The beginning .................................. 3
1.2 Overview ..................................... 3

2 Usage 4

2.1 Process options .................................. 4
  2.1.1 \texttt{ProcessKeyvalOptions} ................. 4
  2.1.2 \texttt{ProcessLocalKeyvalOptions} .......... 4
  2.1.3 \texttt{SetupKeyvalOptions} .................. 4

2.2 Option declarations ............................. 5
  2.2.1 \texttt{DeclareStringOption} .................. 5
  2.2.2 \texttt{DeclareBoolOption} ................... 6
  2.2.3 \texttt{DeclareComplementaryOption} ....... 6
  2.2.4 \texttt{DeclareVoidOption} .................. 7
  2.2.5 \texttt{DeclareDefaultOption} ............... 7
  2.2.6 Local options ............................... 7
  2.2.7 Dynamic options ............................. 7
  2.2.8 \texttt{DisableKeyvalOption} ................. 8
  2.2.9 \texttt{AddToKeyvalOption} .................. 9

2.3 Global vs. local options ....................... 9

2.4 Summary of internal macros ................... 9
2.5 plain \LaTeX{} .................................. 10

3 Example 10

4 Package options 12

4.1 Package \texttt{kvoptions-patch} ................. 12
4.2 Option \texttt{debugshow} ........................ 13

5 Limitations 14

5.1 Compatibility .................................. 14
  5.1.1 Package \texttt{kvoptions-patch} vs. package \texttt{xkvltxp} ......... 14
5.2 Limitations .................................... 14
  5.2.1 Option comparisons .......................... 14
  5.2.2 Option list parsing with package \texttt{kvoptions-patch} ............ 14

\textsuperscript{*}Please report any issues at \url{https://github.com/ho-tex/kvoptions/issues}
# Implementation

6.1 Preamble ................................................. 15
6.2 Option declaration macros ............................... 18
  6.2.1 \SetupKeyvalOptions ............................... 18
  6.2.2 \DeclareBoolOption ................................. 19
  6.2.3 \DeclareStringOption ............................... 21
  6.2.4 \DeclareVoidOption ................................. 22
  6.2.5 \DeclareDefaultOption ............................. 23
  6.2.6 \DeclareLocalOptions .............................. 23
6.3 Dynamic options ........................................ 23
  6.3.1 \DisableKeyvalOption ............................... 23
6.4 Change option code .................................... 26
  6.4.1 \AddToKeyvalOption ................................. 26
6.5 Process options ....................................... 26
  6.5.1 Get global options .................................. 26
  6.5.2 \ProcessKeyvalOptions ............................. 27
  6.5.3 \ProcessLocalKeyvalOptions ....................... 30
  6.5.4 Helper macros ...................................... 31
6.6 plain TEX .............................................. 32
6.7 Package kvoptions-patch ............................... 32

# Test

7.1 Preface for standard catcode check ................... 40
7.2 Catcode checks for loading ............................ 40

# Installation

8.1 Download .............................................. 44
8.2 Bundle installation .................................... 45
8.3 Package installation ................................... 45
8.4 Refresh file name databases ......................... 45
8.5 Some details for the interested ...................... 45

# References

9 References ................................................ 46

# History

10 History .................................................. 46
  [0000/00/00 v0.0] ..................................... 46
  [2004/02/22 v1.0] .................................... 46
  [2006/02/16 v2.0] .................................... 46
  [2006/02/20 v2.1] .................................... 47
  [2006/06/01 v2.2] .................................... 47
  [2006/08/17 v2.3] .................................... 47
  [2006/08/22 v2.4] .................................... 47
  [2007/04/11 v2.5] .................................... 47
  [2007/05/06 v2.6] .................................... 47
  [2007/06/11 v2.7] .................................... 47
  [2007/10/02 v2.8] .................................... 47
  [2007/10/11 v2.9] .................................... 47
  [2007/10/18 v3.0] .................................... 47
  [2009/04/10 v3.1] .................................... 47
  [2009/07/17 v3.2] .................................... 48
  [2009/07/21 v3.3] .................................... 48
  [2009/08/13 v3.4] .................................... 48
  [2009/12/04 v3.5] .................................... 48
  [2009/12/08 v3.6] .................................... 48
Introduction

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 The beginning

This package \texttt{kvoptions} addresses class or package writers that want to allow their users to specify options as key value pairs, e.g.:

\begin{verbatim}
\documentclass[verbose=false,name=me]{myclass}
\usepackage[format=print]{mylayout}
\end{verbatim}

Prominent example is package \texttt{hyperref}, probably the first package that offers this service. It’s \texttt{ProcessOptionsWithKV} is often copied und used in other packages, e.g. package \texttt{helvet} that uses this interface for its option \texttt{scaled}.

However copying code is not the most modern software development technique. And \texttt{hyperref’s} code for \texttt{ProcessOptionsWithKV} was changed to fix bugs. The version used in other packages depends on the time of copying and the awareness of \texttt{hyperref’s} changes. Now the code is sourced out into this package and available for other package or class writers.

1.2 Overview

Package \texttt{kvoptions} connects package \texttt{keyval} with \LaTeX’s package and class \texttt{options}:

<table>
<thead>
<tr>
<th>Package \texttt{keyval}</th>
<th>Package \texttt{kvoptions}</th>
<th>\LaTeX kernel</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{\define@key}</td>
<td>\texttt{\DeclareVoidOption}</td>
<td>\texttt{\DeclareOption}</td>
</tr>
<tr>
<td></td>
<td>\texttt{\DeclareStringOption}</td>
<td></td>
</tr>
<tr>
<td></td>
<td>\texttt{\DeclareBoolOption}</td>
<td></td>
</tr>
<tr>
<td></td>
<td>\texttt{\DeclareComplementaryOption}</td>
<td></td>
</tr>
<tr>
<td></td>
<td>\texttt{\DisableKeyvalOption}</td>
<td></td>
</tr>
<tr>
<td></td>
<td>\texttt{\DeclareDefaultOption}</td>
<td>\texttt{\DeclareOption*}</td>
</tr>
<tr>
<td>\texttt{ProcessKeyvalOptions}</td>
<td>\texttt{\ProcessOptions*}</td>
<td></td>
</tr>
<tr>
<td>Option patch</td>
<td>Class/package option system</td>
<td></td>
</tr>
<tr>
<td>\texttt{\SetupKeyvalOptions}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2 Usage

2.1 Process options

2.1.1 \ProcessKeyvalOptions

\ProcessKeyvalOptions{(family)} \ProcessKeyvalOptions*

This command evaluates the global or local options of the package that are defined with keyval's interface within the family \texttt{(family)}. It acts the same way as \LaTeX's \texttt{\ProcessOptions*}. In a package unknown global options are ignored, in a class they are added to the unknown option list. The known global options and all local options are passed to keyval's \texttt{\setkeys} command for executing the options. Unknown options are reported to the user by an error.

If the family name happens to be the same as the name of the package or class where \texttt{\ProcessKeyvalOptions} is used or the family name has previously been setup by \texttt{\SetupKeyvalOptions}, then \texttt{\ProcessKeyvalOptions} knows the family name already and you can use the star form without mandatory argument.

2.1.2 \ProcessLocalKeyvalOptions

\ProcessLocalKeyvalOptions{(family)} \ProcessLocalKeyvalOptions*

This macro has the same syntax and works similar as \texttt{\ProcessKeyvalOptions}. However it ignores global options and only processes the local package options. Therefore it only can be used inside a package. An error is thrown, if it is used inside a class.

Neither of the following macros are necessary for \texttt{\ProcessKeyvalOptions}. They just help the package/class author in common tasks.

2.1.3 \SetupKeyvalOptions

\SetupKeyvalOptions{
    family = (family),
    prefix = (prefix),
    setkeys = (setkeys command)
}

This command allows to configure the default assumptions that are based on the current package or class name. \LaTeX remembers this name in \texttt{@currname}. The syntax description of the default looks a little weird, therefore an example is given for a package or class named \texttt{foobar}.
Key setkeys was added in version 3.9. The original \setkeys of package keyval is not reentrant. If an option is processed by this \setkeys, then the option should not call \setkeys again with a different family. Otherwise the next options of the first \setkeys call are processed with the wrong family. With key setkeys the macro \kvsetkeys can be set that does not have the problem of the original \setkeys of package keyval.

Probably \setkeys of package xkeyval is safe in this respect. But I haven’t made a full analysis. At least it does not have the problem of the original \setkeys.

### 2.2 Option declarations

The options for \ProcessKeyvalOptions are defined by keyval’s \define@key. Common purposes of such keys are boolean switches, they enable or disable something. Or they store a name or some kind of string in a macro. The following commands help the user. He declares what he wants and \kvoptions take care of the key definition, resource allocation and initialization.

In order to avoid name clashes of macro names, internal commands are prefixed. Both the prefix and the family name for the defined keys can be configured by \SetupKeyvalOptions.

#### 2.2.1 \DeclareStringOption

```
\DeclareStringOption [(init)] {⟨key⟩} [(default)]
```

A macro is created that remembers the value of the key ⟨key⟩. The name of the macro consists of the option name ⟨key⟩ that is prefixed by the prefix (see 2.1.3). The initial contents of the macro can be given by the first optional argument ⟨init⟩. The default is empty.

The the option ⟨key⟩ is defined. The option code just stores its value in the macro. If the optional argument at the end of \DeclareStringOption is given, then option ⟨key⟩ is defined with the default ⟨default⟩.

Example for a package with the following two lines:

\ProvidesPackage{foobar}
\DeclareStringOption[me]{name}

Then \DeclareStringOption defines the macro with content me, note LATEX complains if the name of the macro already exists:

\newcommand*{\foobar@name}{me}

The option definition is similar to:

\begin{verbatim}
\define@key{foobar}{name}\
\renewcommand*{\foobar@name}{#1}\
\end{verbatim}
2.2.2 \DeclareBoolOption

\DeclareBoolOption [(init)]{(key)}

A boolean switch is generated, initialized by value \textit{(init)} and the corresponding key \textit{(key)} is defined. If the initialization value is not given, \texttt{false} is used as default.

The internal actions of \texttt{\DeclareBoolOption} are shown below. The example is given for a package author who has the following two lines in his package/class:

\begin{verbatim}
\ProvidesPackage{foobar}
\DeclareBoolOption{verbose}
\end{verbatim}

First a new switch is created:

\begin{verbatim}
\newif\iffoobar@verbose
\end{verbatim}

and initialized:

\begin{verbatim}
\foobar@verbosefalse
\end{verbatim}

Finally the key is defined:

\begin{verbatim}
\define@key{foobar}{verbose}{true}{...}
\end{verbatim}

The option code configures the boolean option in the following way: If the author specifies \texttt{true} or \texttt{false} then the switch is turned on or off respectively. Also the option can be given without explicit value. Then the switch is enabled. Other values are reported as errors.

Now the switch is ready to use in the package/class, e.g.:

\begin{verbatim}
\ifeboobar@verbose
  \% print verbose message
\else
  \% be quiet
\fi
\end{verbatim}

Users of package \texttt{\ifthen} can use the switch as boolean:

\begin{verbatim}
\boolean{foobar@verbose}
\end{verbatim}

2.2.3 \DeclareComplementaryOption

\DeclareComplementaryOption {{key}} {{parent}}

Sometimes contrasting names are used to characterize the two states of a boolean switch, for example \texttt{draft} vs. \texttt{final}. Both options behave like boolean options but they do not need two different switches, they should share one. \texttt{\DeclareComplementaryOption} allows this. The option \textit{(key)} shares the switch of option \textit{(parent)}. Example:

\begin{verbatim}
\DeclareBoolOption{draft}
\DeclareComplementaryOption{final}{draft}
\end{verbatim}

Then \texttt{final} sets the switch of \texttt{draft} to \texttt{false}, and \texttt{final=false} enables the \texttt{draft} switch.
2.2.4 \DeclareVoidOption

\DeclareVoidOption\{{key}\}\{{code}\}

\ProcessKeyvalOptions can be extended to recognize options that are declared in traditional way by \DeclareOption. But in case of the error that the user specifies a value, then this option would not recognized as key value option because of \DeclareOption and not detected as traditional option because of the value part. The user would get an unknown option error, difficult to understand.

\DeclareVoidOption solves this problem. It defines the option \{key\} as key value option. If the user specifies a value, a warning is given and the value is ignored.

The code part \{code\} is stored in a macro. The name of the macro consists of the option name \{key\} that is prefixed by the prefix (see 2.1.3). If the option is set, the macro will be executed. During the execution \CurrentOption is available with the current key name.

2.2.5 \DeclareDefaultOption

\DeclareDefaultOption\{{code}\}

This command does not define a specific key, it is the equivalent to \LaTeX’s \DeclareOption*. It allows the specification of a default action \{code\} that is invoked if an unknown option is found. While \{code\} is called, macro \CurrentOption contains the current option string. In addition \CurrentOptionValue contains the value part if the option string is parsable as key value pair, otherwise it is \relax. \CurrentOptionKey contains the key of the key value pair, or the whole option string, if it misses the equal sign.

Inside packages typical default actions are to pass unknown options to another package. Or an error message can be thrown by \@unknownoptionerror. This is the original error message that \LaTeX gives for unkown package options. This error message is easier to understand for the user as the error message from package keyval that is given otherwise.

A Class ignores unknown options and puts them on the unused option list. Let \LaTeX do the job and just call \OptionNotUsed. Or the options can be passed to another class that is later loaded.

2.2.6 Local options

\DeclareLocalOption \{(option)\}
\DeclareLocalOptions \{(option list)\}

Both macros mark package options as local options. That means that they are ignored by \ProcessKeyvalOptions if they are given as global options. \DeclareLocalOptions takes one option, \DeclareLocalOptions expects a comma separated list of options.

2.2.7 Dynamic options

Options of \LaTeX’s package/class system are cleared in \ProcessOptions. They modify the static model of a package. For example, depending on option bookmarks package hyperref loads differently.
Options, however, defined by \texttt{keyval}'s \texttt{\define@key} remain defined, if the options are processed by \texttt{\setkeys}. Therefore these options can also be used to model the dynamic behaviour of a package. For example, in \texttt{hyperref} the link colors can be changed everywhere until the end in \texttt{\end{document}}.

However package \texttt{color} that adds color support is necessary and it cannot be loaded after \texttt{\begin{document}}. Option \texttt{colorlinks} that loads \texttt{color} should be active until \texttt{\begin{document}} and die in some way if it is too late for loading packages. With \texttt{\DisableKeyvalOption} the package/class author can specify and configure the death of an option and controls the life period of the option.

### 2.2.8 \texttt{\DisableKeyvalOption}

\begin{verbatim}
\DisableKeyvalOption [[⟨options⟩]] ⟨⟨family⟩⟩ ⟨⟨key⟩⟩
⟨⟨options⟩⟩:
  action = undef, warning, error, or ignore default: undef
  global or local default: global
  package or class = ⟨⟨name⟩⟩
\end{verbatim}

\texttt{\DisableKeyvalOption} can be called to mark the end when the option \texttt{⟨⟨key⟩⟩} is no longer useful. The behaviour of an option after its death can be configured by action:

- **undef**: The option will be undefined, If it is called, \texttt{\setkeys} reports an error because of unknown key.
- **error or warning**: Use of the option will cause an error or warning message. Also these actions require that exclusively either the package or class name is given in options package or class.
- **ignore**: The use of the option will silently be ignored.

The option's death can be limited to the end of the current group, if option \texttt{local} is given. Default is \texttt{global}.

The package/class author can wish the end of the option already during the package loading, then he will have static behaviour. In case of dynamic options \texttt{\DisableKeyvalOption} can be executed everywhere, also outside the package. Therefore the family name and the package/class name is usually unknown for \texttt{\DisableKeyvalOption}. Therefore the argument for the family name is mandatory and for some actions the package/class name must be provided.

Usually a macro would configure the option death, Example:

\begin{verbatim}
\ProvidesPackage{foobar}
\DeclareBoolOption{color}
\DeclareStringOption[red]{emphcolor}
\ProcessKeyvalOptions*

\newcommand*{\foobar@DisableOption}[2]{{% \DisableKeyvalueOption[
  action=#1],
  package=foobar
 }{\foobar}{#2}%}

\iffoobar@color
\RequirePackage{color}
\renewcommand*{\emp}[1]{\textcolor{\foobar@emphcolor}{#1}}
\end{verbatim}
\else
  \% Option emphcolor is not wrong, if we have color support.
  \% otherwise the option has no effect, but we don’t want to
  \% remove it. Therefore action ‘ignore’ is the best choice:
  \foobar@DisableOption{ignore}{emphcolor}
\fi
\% No we don’t need the option 'color'.
\foobar@DisableOption{warning}{color}
\%
\% With color support option 'emphcolor' will dynamically
\% change the color of \texttt{emph} statements.

\subsection{AddToKeyvalOption}

\begin{verbatim}
\AddToKeyvalOption{{⟨family⟩}}{{⟨key⟩}}{{⟨code⟩}}
\AddToKeyvalOption*{{⟨key⟩}}{{⟨code⟩}}
\end{verbatim}

The code for an existing key ⟨key⟩ of family ⟨family⟩ is extended by code ⟨code⟩. In
the starred form the current family setting is used, see \texttt{\ProcessKeyvalOptions*}.

\section{Global vs. local options}

Options that are given for \texttt{\documentclass} are called global options. They are
known to the class and all packages. A package may make use of a global option
and marks it as used. The advantage for the user is the freedom to specify options
both in the \texttt{\documentclass} or \texttt{\usepackage} commands.

However global options are shared with the class options and options of all
other packages. Thus there can be the same option with different semantics for
different packages and classes. As example, package \texttt{bookmark} knows option \texttt{open}
that specifies whether the bookmarks are opened or closed initially. It’s values are
\texttt{true} or \texttt{false}. Since KOMA-Script version 3.00 the KOMA classes also introduces
option \texttt{open} with values \texttt{right} and \texttt{any} and a complete different meaning.

Such conflicts can be resolved by marking all or part of options as local by
\texttt{\DeclareLocalOption} or \texttt{\DeclareLocalOptions}. Then the packages ignores
global occurrences of these options. Package \texttt{kvoptions} provides two methods:

\begin{itemize}
  \item \texttt{\ProcessLocalKeyvalOptions} automatically uses all options as local options.
        It ignores all global options.
  \item \texttt{\DeclareLocalOption} or \texttt{\DeclareLocalOptions} marks options as local options.
        \texttt{\ProcessKeyvalOptions} will then ignore global occurrences for these local options.
\end{itemize}

Since version 1.5 package \texttt{bookmark} uses the latter method. It checks global and
local option places for driver options and limits all other options as local options.
Thus the class option \texttt{open} of KOMA-Script is not misread as option for package
\texttt{bookmark}.

\section{Summary of internal macros}

The \texttt{\Declare...Option} commands define macros, additionally to the macros
generated by the key definition. These macros can be used by the package/class
author. The name of the macros starts with the prefix ⟨prefix⟩ that can be con-
figured by \texttt{\SetupKeyvalOptions}. 

2.5 plain \TeX

Package keyval is also usable in plain \TeX{} with the help of file miniltx.tex. Some features of this package kvoptions might also be useful for plain \TeX{}. If \LaTeX{} is not found, \ProcessKeyvalOptions{} and option patch are disabled. Before using the option declaration commands \Declare...
Option, \SetupKeyvalOptions{} must be used.

3 Example

The following example defined a package that serves some private color management. A boolean option print enables print mode without colors. An option emph redefines \texttt{\textbf{} to print in the given color. And the driver can be specified by option driver.

\begin{verbatim}
1 (*example*)
2 \% Package identification
3 \% -----------------------
4 \NeedsTeXFormat{LaTeX2e}
5 \ProvidesPackage{example-mycolorsetup}[2019/11/29 Managing my colors]
6
7 \RequirePackage{iftex}
8 \RequirePackage{kvoptions}
9
10 \% Option declarations
11 \% ---------------------
12
13 \SetupKeyvalOptions{
14 family=MCS,
15 prefix=MCS@
16 }
17 \% Use a shorter family name and prefix
18
19 \% Option print
20 \DeclareBoolOption{print}
21 \% is the same as
22 \% \DeclareBoolOption[false]{print}
23
24 \% Option driver
25 \ifpdf
26 \DeclareStringOption[pdftex]{driver}
27 \else
28 \DeclareStringOption[dvips]{driver}
29 \fi
30
31 \% Alternative interface for driver options
\end{verbatim}

\begin{tabular}{|l|l|l|}
\hline
Declare \texttt{(key)} & Defined macro & Description \\
\hline
\texttt{\textbackslash DeclareStringOption} & \texttt{\langle prefix\rangle\langle key\rangle} & holds the string \\
\texttt{\textbackslash DeclareBoolOption} & \texttt{\langle prefix\rangle\langle key\rangle false} & boolean switch \\
 & \texttt{\langle prefix\rangle\langle key\rangle true} & disable switch \\
\texttt{\textbackslash DeclareComplementaryOption} & \texttt{\langle prefix\rangle\langle key\rangle false} & enable parent switch \\
 & \texttt{\langle prefix\rangle\langle key\rangle true} & disable parent switch \\
\texttt{\textbackslash DeclareVoidOption} & \texttt{\langle prefix\rangle\langle key\rangle} & holds the action \\
\hline
\end{tabular}
\DeclareVoidOption{dvips}{\SetupDriver}
\DeclareVoidOption{dvipdfm}{\SetupDriver}
\DeclareVoidOption{pdftex}{\SetupDriver}
% In \SetupDriver we take the current option \CurrentOption
% and pass it to the driver option.
% The \expandafter commands expand \CurrentOption at the
% time, when \SetupDriver is executed and \CurrentOption
% has the correct meaning.
\newcommand*{%\SetupDriver}{%\expandafter\@SetupDriver\expandafter{\CurrentOption}{}%
\newcommand*{%\@SetupDriver}{%[1]{%}
\setkeys{MCS}{driver={#1}{%}
\newcommand*{%}{%}
% Option emph
% An empty value means, we want to have no color for \emph.
% If the user specifies option emph without value, the red is used.
\DeclareStringOption{%\emph}{[red]}
% is the same as
% \DeclareStringOption{%}{emph}{[red]}
\DeclareDefaultOption{%}
% Default option rule
% \PackageWarningNoLine{%currname}{% Unknown option \CurrentOption\MessageBreak
% is passed to package ‘color’%}
% Pass the option to package color.
% Again it is better to expand \CurrentOption.
% \PassOptionsToPackage\expandafter{\CurrentOption}{color}%
% Package color does not take options with values.
% We provide the standard LaTeX error.
% \@unknownoptionerror
\if\CurrentOptionValue\relax
% PackageWarningNoLine{%currname}{% Unknown option \CurrentOption\MessageBreak
% is passed to package ‘color’%}
% Pass the option to package color.
% Again it is better to expand \CurrentOption.
% \PassOptionsToPackage\expandafter{\CurrentOption}{color}%
% Package color does not take options with values.
% We provide the standard LaTeX error.
% \@unknownoptionerror
\fi
}
% Process options
% -------------------
% Implementation depending on option values
% Code for print mode
\ifMCS@print
\PassOptionsToPackage{monochrome}{color}%
\fi
\RequirePackage{%MCS@driver}{color}%
% load package color with the correct driver
\if\MCS@emph\relax
\empty
\fi
% \empty is a predefined macro with empty contents.
4 Package options

The package `kvoptions` knows two package options `patch` and `debugshow`. The options of package `kvoptions` are intended for authors, not for package/class writers. Inside a package it is too late for option `patch` and `debugshow` enables some messages that are perhaps useful for the debugging phase. Also \TeX is unhappy if a package is loaded later again with options that are previously not given. Thus package and class authors, stay with `\RequirePackage{kvoptions}` without options.

Option `patch` loads package `kvoptions-patch`.

4.1 Package `kvoptions-patch`

\TeX's system of package/class options has some severe limitations that especially affects the value part if options are used as pair of key and value.

- Spaces are removed, regardless where:
  \begin{verbatim}
  \documentclass[box=0 0 400 600]{article}
  \end{verbatim}
  Now each package will see `box=00400600` as global option.

- In the previous case also braces would not help:
  \begin{verbatim}
  \documentclass[box={0 0 400 600}]{article}
  \end{verbatim}
  The result is an error message:
  \begin{verbatim}
  ! \TeX Error: Missing \begin{document}.
  \end{verbatim}
  As local option, however, it works if the package knows about key value options (By using this package, for example).

- The requirements on robustness are extremly high. \TeX expands the option. All that will not work as environment name will break also as option. Even a `\relax` will generate an error message:
  \begin{verbatim}
  ! Missing \endcsname inserted.
  \end{verbatim}
  Of course, \TeX does not use its protecting mechanisms. On contrary `\protect` itself will cause errors.

- The options are expanded. But perhaps the package will do that, because it has to setup some things before? Example `hyperref`:
  \begin{verbatim}
  \usepackage[pdauthor=M"uller]{hyperref}
  \end{verbatim}
  Package `hyperref` does not see `M"uller` but its expansion and it does not like it, you get many warnings.
And the title becomes: Müller. Therefore such options must usually be given after package hyperref is loaded:

\usepackage{hyperref}
\hypersetup{pdfauthor=Francoise Muller}

As package option it will even break with Francoise because of the cedilla \c c, it is not robust enough.

For users that do not want with this limitations the package offers package kvoptions-patch. It patches \LaTeX{}'s option system and tries to teach it also to handle options that are given as pairs of key and value and to prevent expansion. It can already be used at the very beginning, before \documentclass:

\begin{verbatim}
\RequirePackage{kvoptions-patch}
\documentclass[\pdfauthor=Francoise Muller]{article}
\usepackage{hyperref}
\end{verbatim}

The latest time is before the package where you want to use problematic values:

\begin{verbatim}
\usepackage{kvoptions-patch}
\usepackage[Francoise Muller]{hyperref}
\end{verbatim}

Some remarks:

- The patch requires \epsilon-\TeX, its \texttt{unexpanded} feature is much too nice. It is possible to work around using token registers. But the code becomes longer, slower, more difficult to read and maintain. The package without option \texttt{patch} works and will work without \epsilon-\TeX.

- The code for the patch is quite long, there are many test cases. Thus the probability for bugs is probably not too small.

- Since 2008/10/18 v3.0 package kvoptions-patch is available. Before option \texttt{patch} of package kvoptions must be used instead. I think, the solution as standalone package kvoptions-patch is cleaner and avoids option clashes.

### 4.2 Option debugshow

The name of this option follows the convention of packages multicol, tabularx, and tracefnt. Currently it prints the setting of boolean options, declared by \texttt{\DeclareBoolOption} in the .log file, if that boolean option is used. You can activate the option by:

- \texttt{\PassOptionsToPackage{debugshow}{kvoptions}}
  Put this somewhere before package kvoptions is loaded first, e.g. before \texttt{\documentclass}.

- \texttt{\RequirePackage{debugshow}{kvoptions}}
  Before \texttt{\documentclass} even an author has to use \texttt{\RequirePackage}. \texttt{\usepackage} only works after \texttt{\documentclass}.

The preferred method is \texttt{\PassOptionsToPackage}, because it does not force the package loading and does not disturb, if the package is not loaded later at all.
5 Limitations

5.1 Compatibility

5.1.1 Package `kvoptions-patch` vs. package `xkvltxp`

Package `xkvltxp` from the `xkeyval` project has the same goal as `kvoptions-patch` and to patch LaTeX’s kernel commands in order to get better support for key value options. Of course they cannot be used both. The user must decide, which method he prefers. Package `kvoptions-patch` aborts itself, if it detects that `xkvltxp` is already loaded.

However package `xkvltxp` and `kvoptions` can be used together, example:

```
\usepackage{xkvltxp}
\usepackage[...]{foobar} % foobar using kvoptions
```

The other way should work, too.

Package `kvoptions-patch` tries to catch more situations and to be more robust.

5.2 Limitations

5.2.1 Option comparisons

In some situations LaTeX compares option lists, e.g. option clash check, `\@ifpackagewith`, or `\@ifclasswith`. Apart from catcode and sanitizing problems of option patch, there is another problem. LaTeX does not know about the type and default values of options in key value style. Thus an option clash is reported, even if the key value has the same meaning:

```
\usepackage[scaled=0.7]{helvet}
\usepackage[scaled = 0.7]{helvet}
```

5.2.2 Option list parsing with package `kvoptions-patch`

With package `kvoptions-patch` the range of possible values in key value specifications is much large, for example the comma can be used, if enclosed in curly braces.

Other packages, especially the packages that uses their own process option code can be surprised to find tokens inside options that they do not expect and errors would be the consequence. To avoid errors the options, especially the unused option list is sanitized. That means the list will only contain tokens with catcode 12 (other) and perhaps spaces (catcode 10). This allows a safe parsing for other packages. But a comma in the value part is no longer protected by curly braces because they have lost their special meaning. This is the price for compatibility.

Example:
\RequirePackage{kvoptions-patch}
\documentclass[a={a,b,c},b]{article}
\begin{document}
\end{document}

Result:

LaTeX Warning: Unused global option(s):
   \[a={a,c},b\].

6 Implementation

6.1 Preamble

\texttt{\LaTeX} Warning: Unused global option(s):
   \[a={a,c},b\].

Reload check and identification.  Reload check, especially if the package is
not used with \LaTeXX.

\begin{verbatim}
\begingroup\catcode61\catcode48\catcode32=10\relax%
\catcode13=5 % ^^M
\endlinechar=13 %
\catcode35=6 % #
\catcode39=12 % '
\catcode40=12 % (
\catcode41=12 % )
\expandafter\let\expandafter\x\csname ver@kvoptions.sty\endcsname
\ifx\x\relax % plain-TeX, first loading
\else
\def\empty{}%
\ifx\x\empty % LaTeX, first loading,
% variable is initialized, but \ProvidesPackage not yet seen
\else
\expandafter\ifx\csname PackageInfo\endcsname\relax
\def\x#1#2{\immediate\write-1{Package #1 Info: #2.}}%
\else
\def\x#1#2{\PackageInfo{#1}{#2, stopped}}%
\fi
\x{kvoptions}{The package is already loaded}%
\aftergroup\endinput
\fi
\fi\endgroup%
\end{verbatim}

Package identification:

\begin{verbatim}
\begingroup\catcode61\catcode48\catcode32=10\relax%
\catcode13=5 % ^^M
\endlinechar=13 %
\catcode35=6 % #
\catcode39=12 % '
\catcode40=12 % (}
\end{verbatim}

External resources. The package extends the support for key value pairs of package \keyval to package options. Thus the package needs to be loaded anyway and we use it for \SetupKeyvalOptions. AFAIK this does not disturb users of xkeyval.

Macro \DeclareLocalOptions parses a comma separated key list and uses \comma@parse of package kvsetkeys, version 1.3.

Provide macros for plain \TeX.

Options Option debugshow enables additional lines of code that prints information into the .log file.
6.2 Option declaration macros

6.2.1 \SetupKeyvalOptions

The family for the key value pairs can be setup once and is remembered later. The package name seems a reasonable default for the family key, if it is not set by the package author.

\KVO@family

We cannot store the family setting in one macro, because the package should be usable for many other packages, too. Thus we remember the family setting in a macro, whose name contains the package name with extension, a key in \LaTeX's class/package system.

\define@key{KVO}{family}{% \
\expandafter\edef\csname KVO@family@\@currname.\@currext\endcsname{#1}% 
\}
\def\KVO@family{\@ifundefined{KVO@family@\@currname.\@currext}{\@currname}{\csname KVO@family@\@currname.\@currext\endcsname}}

\KVO@prefix

The value settings of options that are declared by \DeclareBoolOption and \DeclareStringOption need to be saved in macros. In the first case this is a switch \if\langle prefix\rangle\langle key\rangle, in the latter case a macro \langle prefix\rangle\langle key\rangle. The prefix can be configured, by prefix that is declared here. The default is the package name with @ appended.

\define@key{KVO}{prefix}{% \
\expandafter\edef\csname KVO@prefix@\@currname.\@currext\endcsname{#1}% 
\}
\def\KVO@prefix{\ltx@ifundefined{KVO@prefix@\@currname.\@currext}{\@currname @}{\csname KVO@prefix@\@currname.\@currext\endcsname}}

\KVO@setkeys

\define@key{KVO}{setkeys}{% \
\expandafter\def\csname KVO@setkeys@\@currname.\@currext\endcsname{#1}% 
\}
\def\KVO@setkeys{\ltx@IfUndefined{KVO@setkeys@\@currname.\@currext}{\setkeys}{\csname KVO@setkeys@\@currname.\@currext\endcsname}}

18
\SetupKeyvalOptions

The argument of \SetupKeyvalOptions expects a key value list, known keys are
family and prefix.

\DeclareBoolOption

Usually options of boolean type can be given by the user without value and this
means a setting to true. We follow this convention here. Also it simplifies the user
interface.

The switch is created and initialized with false. The default setting can be
overwritten by the optional argument.

\DeclareComplementaryOption

The first argument is the key name, the second the key that must be a boolean
option with the same current family and prefix. A new switch is not created for
the new key, we have already a switch. Instead we define switch setting commands
to work on the parent switch.
\newcommand*{\DeclareComplementaryOption}[2]{%
  \ifundefined{if\KVO@prefix#2}{%
    \PackageError{kvoptions}{% Cannot generate option code for ‘#1’, \MessageBreak parent switch ‘#2’ does not exist%}{%
      You are inside %
      \ifx\@currext\@clsextension class\else package\fi\space
      ‘\@currname.\@currext’. \MessageBreak
      ‘\KVO@family’ is used as family % for the keyval options. \MessageBreak
      ‘\KVO@prefix’ serves as prefix % for internal switch macros. \MessageBreak
    \MessageBreak
    \@ehc
  }{%}
  }{%}
  \KVO@ifdefinable{\KVO@prefix#1true}{%
    \KVO@ifdefinable{\KVO@prefix#1false}{%
      \expandafter\let\csname\KVO@prefix#1false\expandafter\endcsname
      \csname\KVO@prefix#2true\endcsname
      \expandafter\let\csname\KVO@prefix#1true\expandafter\endcsname
      \csname\KVO@prefix#2false\endcsname
      The same code part as in \DeclareBoolOption can now be used.
      \begingroup
      \edef\x{\endgroup
        \noexpand\define@key{\KVO@family}{#1}[true]{%\noexpand\KVO@boolkey{\@currname}
          \ifx\@currext\@clsextension
            \noexpand\@clsextension
          \else
            \noexpand\@pkgextension
          \fi
          \{\KVO@prefix}{#1}{####1}%
      }%
      \x
    }%
  }{%}
}\KVO@ifdefinable \秦维@ifdefinable Generate the command token LaTeX’s \ifdefinable expects.
\def\KVO@ifdefinable#1{% \noexpand\define@key{\KVO@family}{#1}[true]{%
  \noexpand\KVO@boolkey{\@currname}%
  \ifx\@currname\@clsextension
    \noexpand\@clsextension
  \else
    \noexpand\@pkgextension
  \fi
  \{\KVO@prefix}{#1}{####1}%
}\x
}\x

\KVO@boolkey We check explicitly for true and false to prevent the user from accidently calling other macros.

#1 package/class name
#2 \@pkgextension/\@clsextension
#3 prefix
#4 key name
#5 new value
\def\KVO@boolkey#1#2#3#4#5{% \edef\KVO@param{#5}%
The macros `\KVO@true` and `\KVO@false` are used for string comparisons. After `\ltx@onelevel@sanitize` we have only tokens with catcode 12 (other).

```latex
\def\KVO@true{true}
\def\KVO@false{false}
\ltx@onelevel@sanitize\KVO@true
\ltx@onelevel@sanitize\KVO@false
```

6.2.3 \texttt{\DeclareStringOption}

```
\newcommand*{\DeclareStringOption}[2][]{%\iffnextchar[\KVO@DeclareStringOption{\iffnextchar[]{\{\iffnextchar\%\KVO@DeclareStringOption[2]{}\endcsname\endgroup\iffnextchar\}{\{}\iffnextchar\%\KVO@DeclareStringOption[2]{}}}\endcsname\endgroup}\iffnextchar[\KVO@DeclareStringOption[2]{}\endcsname\endgroup}
```

\texttt{\KVO@DeclareStringOption}

```latex
\def\KVO@DeclareStringOption#1#2#3[#4]{%\iffnextchar[\KVO@ifdefined{\KVO@prefix#2}{%\begingroup\iffnextchar\%\KVO@DeclareStringOption[2]{}\endcsname\endgroup\iffnextchar\}{\{}\iffnextchar\%\KVO@DeclareStringOption[2]{}}}\endcsname\endgroup}
```

\texttt{\KVO@ifdefined}
\edef\x{\endgroup
\noexpand\define@key{\KVO@family}{#2}{\the\toks@{%
^^A \begingroup
^^A \toks@{####1}%
^^A \ifx\@currext\@clsextension
^^A \noexpand\ClassInfo
^^A \else
^^A \noexpand\PackageInfo
^^A \fi
^^A \{\@currname}{%}
^^A [option] #2={\noexpand\the\toks0}%
^^A }%
^^A \endgroup
\noexpand\def
\expandafter\noexpand\csname\KVO@prefix#2\endcsname{####1}%
}\}%
\x
}\}%
\x
}\}%
}

6.2.4 \DeclareVoidOption
\DeclareVoidOption
\newcommand*{\DeclareVoidOption}[2]{% \egingroup
\let\next@\gobbletwo
\KVO@ifdefinable{\KVO@prefix#1}{%
\let\next@\firstofone
}% 
\expandafter\endgroup
\next{%
\begingroup
\edef\x{\endgroup
\noexpand\define@key{\KVO@family}{#1}{\KVOVOID@}{% 
\noexpand\KVOVOIDkey{\@currname}%
\ifx\@currext\@clsextension
\noexpand\@clsextension
\else
\noexpand\@pkgextension
\fi
[#1]%
[###1]%
\expandafter\endgroup
\noexpand\csname\KVO@prefix#1\endcsname
}\}%
\x
\begingroup
\toks@{#2}%
\expandafter\endgroup
\expandafter\def
\csname\KVOVOID@\endcsname
\begin{verbatim}
\def\KVO@voidkey#1#2#3#4{%
\def\CurrentOption{#3}%
\begingroup
\def\x{#4}%
\expandafter\endgroup
\ifx\x\KVO@VOID@
\else
\ifx#2\@clsextension
\expandafter\ClassWarning
\else
\expandafter\PackageWarning
\fi
\fi
\^^A\ifx#2\@clsextension
\^^A \expandafter\ClassInfo
\else
\^^A \expandafter\PackageInfo
\fi
\^^A{#1}{[option] #3}%
}
\end{verbatim}

\section{Dynamic options}
\subsection{DisableKeyvalOption}
\begin{verbatim}
\SetupKeyvalOptions{%
family=KVOdyn,%
prefix=KVOdyn@%
}\end{verbatim}

\subsection{DeclareDefaultOption}
\begin{verbatim}
\newcommand*{\DeclareDefaultOption}{%
\@namedef{KVO@default@\@currname.\@currext}%
\end{verbatim}

\subsection{DeclareLocalOptions}
\begin{verbatim}
\newcommand*{\DeclareLocalOptions}{[1]{% 
\comma@parse{#1}\KVO@DeclareLocalOption
\end{verbatim}

\subsection{Dynamic options}
\begin{verbatim}
\def\KVO@DeclareLocalOption#1{%
\expandafter\def\csname KVO@local@KVO@family @#1\endcsname{}%
}
\end{verbatim}

\begin{verbatim}
\end{verbatim}

\end{verbatim}
\DeclareBoolOption[true]{global}
\DeclareComplementaryOption{local}{global}
\DeclareStringOption[undef]{action}
\let\KVOdyn@name\relax
\let\KVOdyn@ext\@empty
\define@key{KVOdyn}{class}{%
  \def\KVOdyn@name{#1}%
  \let\KVOdyn@ext\@clsextension
}
\define@key{KVOdyn}{package}{%
  \def\KVOdyn@name{#1}%
  \let\KVOdyn@ext\@pkgextension
}
\newcommand*{\DisableKeyvalOption}[3][]{%
  \begingroup
  \kvsetkeys{KVOdyn}{#1}%
  \def\x{\endgroup}%
  \@ifundefined{KVO@action@#3}{%
    \PackageError{kvoptions}{% Unknown disable action \expandafter\strip@prefix\meaning#3 for option \texttt{#3} in keyval family \texttt{#2}}\@ehc
  }{%
    \csname KVO@action@#3\endcsname{#2}{#3}%
  }\x%
}
\def\KVO@action@undef#1#2{%
  \edef\x{\endgroup
    \ifKVOdyn@global\global\fi
    \let\expandafter\noexpand\csname KV@#1@#2\endcsname\relax
    \ifKVOdyn@global\global\fi
    \let\expandafter\noexpand\csname KV@#1@#2@default\endcsname\relax
  }%
  \PackageInfo{kvoptions}{% \[option] key \texttt{#2} of family \texttt{#1} is disabled (undef, \ifKVOdyn@global global\else local\fi)%
}
\def\KVO@action@ignore#1#2{%
  \edef\x{\endgroup
    \ifKVOdyn@global\global\fi
    \let\expandafter\noexpand\csname KV@#1#2\endcsname\noexpand\@gobble
    \ifKVOdyn@global\global\fi
    \let\expandafter\noexpand\csname KV@#1#2@default\endcsname\noexpand\@empty
  }%
  \PackageInfo{kvoptions}{% \[option] key \texttt{#2} of family \texttt{#1} is disabled (ignore, \ifKVOdyn@global global\else local\fi)%
}
\def\KVO@action@error{\KVO@do@action{error}}
\def\KVO@action@warning{\KVO@do@action{warning}}

#1 error or warning
#2 (family)
#3 (key)
\def\KVO@do@action#1#2#3{\ifx\KVOdyn@name\relax
\PackageError{kvoptions}{Action type ‘#1’ needs package/class name for key ‘#3’ in family ‘#2’}{@ehc}
\else
\edef\x{\endgroup
\noexpand\define@key{#2}{#3}{%\expandafter
\noexpand\csname KVO@disable@#1\endcsname\KVOdyn@ext{#3}}%\ifKVOdyn@global
\global\let\expandafter\noexpand\csname KV@#2@#3\endcsname\expandafter\csname KV@#2@#3@default\endcsname\fi}
\fi}
\def\KVO@disable@error#1#2#3{\ifx#2\@clsextension\expandafter\ClassError
\else\expandafter\PackageError\fi{#1}{Option ‘#3’ is given too late, now the option is ignored}{@ehc}}
\def\KVO@disable@warning#1#2#3{\ifx#2\@clsextension\expandafter\ClassWarning\else\PackageWarning\fi{A [option] key ‘#3’ of family ‘#2’ is disabled (#1, if\KVOdyn@global global \else local)}%}
\def\KVO@disable@error@##1#2#3{%\ifx##2\@clsextension\expandafter\ClassError\else\expandafter\PackageError\fi{##1}{Option ‘#3’ is given too late, now the option is ignored}{@ehc}}
6.4 Change option code

6.4.1 \AddToKeyvalOption

\AddToKeyvalOption

\KVO@AddToKeyvalOption

\KVO@AddToKeyvalOption

\KVO@AddToKeyvalOption

6.5 Process options

6.5.1 Get global options

Package xkeyval removes options with equal signs from the global options (\@classoptionslist). The effect is that other packages and classes will not
see these global options anymore. A bug-report was answered that this behaviour is “by design”. Thus I call it a design bug. Now getting the global options require an algorithm instead of a simple macro call.

\texttt{\KVO@IfDefThen} Call \#2 if command \#1 is defined and not \texttt{\relax}. (Package \texttt{kvoptions\-patch} does not load package \texttt{ltxcmds}.)

If the optional star is given, we get the family name and expand it for safety.

\newcommand*{\ProcessKeyvalOptions}{% 
@ifstar{\begingroup\edef\x{\endgroup\noexpand\KVO@ProcessKeyvalOptions{\KVO@family}}% 
}}% 
\KVO@ProcessKeyvalOptions

\KVO@GetClassOptionsList

Add any global options that are known to KV to the start of the list being built in \KVO@temp and mark them used (by removing them from the unused option list).

```
\ifx\@currext\@clsextension
  \else
  \KVO@GetClassOptionsList
  \ifx\KVO@classoptionslist\relax
    \else
      \@for\KVO@CurrentOption:=\KVO@classoptionslist\do{%
        \@ifundefined{KV@#1@\expandafter\KVO@getkey\KVO@CurrentOption=\@nil}{%}
        \@ifundefined{KVO@local@#1@\expandafter\KVO@getkey\KVO@CurrentOption=\@nil}{%}
        \ifx\KVO@Patch Y%
          \edef\KVO@temp{\etex@unexpanded\expandafter{\KVO@temp},\etex@unexpanded\expandafter{\KVO@CurrentOption},}%
        \else
          \edef\KVO@temp{\KVO@temp,\KVO@CurrentOption,}%
        \fi
        \@expandtwoargs\@removeelement\KVO@CurrentOption\@unusedoptionlist\@unusedoptionlist
      }{}
    \fi
  \fi
\fi
```

Now stick the package options at the end of the list and wrap in a call to \setkeys. A class ignores unknown global options, we must remove them to prevent error messages from \setkeys.

```
\begingroup
  \toks\tw@{}
  \@ifundefined{opt@\@currname.\@currext}{%
    \toks\expandafter{\KVO@temp}
    \etex@unexpanded\expandafter{\KVO@temp}
    ,%
    \etex@unexpanded\expandafter{\KVO@CurrentOption}
    ,%
  }{
    \edef\CurrentOption{\the\toks@}
    \toks\expandafter{\KVO@temp}
    \@for\CurrentOption:=\CurrentOption\do{%
      \@ifundefined{KV@#1@\expandafter\KVO@getkey\CurrentOption=\@nil}{%}
      \@ifundefined{KVO@local@#1@\expandafter\KVO@getkey\CurrentOption=\@nil}{%}
      \ifx\KVO@Patch Y%
        \edef\KVO@temp{\etex@unexpanded\expandafter{\KVO@temp},\etex@unexpanded\expandafter{\KVO@CurrentOption},}%
      \else
        \edef\KVO@temp{\KVO@temp,\KVO@CurrentOption,}%
      \fi
      \@expandtwoargs\@removeelement\KVO@CurrentOption\@unusedoptionlist\@unusedoptionlist
    }{}
  }%
  \endgroup
```

```
\begingroup
  \toks\tw@{}
  \@ifundefined{opt@\@currname.\@current}{%
    \toks\expandafter{\KVO@temp}
    \etex@unexpanded\expandafter{\KVO@temp}
    \etex@unexpanded\expandafter{\csname opt@\@currname.\@current\endcsname}
  }{
    \edef\CurrentOption{\the\toks@}
    \toks\expandafter{\KVO@temp}
    \etex@unexpanded\expandafter{\KVO@CurrentOption}
    \etex@unexpanded\expandafter{\csname opt@\@currname.\@current\endcsname}
  }%
  \@ifundefined{KV@#1@\expandafter\KVO@getkey\CurrentOption=\@nil}{%}
  \@ifundefined{KV@#1@\expandafter\KVO@getkey\CurrentOption=\@nil}{%}
  \edef\KVO@temp{\etex@unexpanded\expandafter{\KVO@temp},\etex@unexpanded\expandafter{\KVO@getkey\CurrentOption},\CurrentOption=\@nil}
\endgroup
```
A class puts not used options in the unused option list unless there is a default handler.

Without default action we pass all options to \setkeys. Otherwise we have to check which options are known. These are passed to \setkeys. For the others the default action is performed.

```latex
\@ifundefined{KVO@default\@currname.\@currext}{%
\@if\KVO@Patch Y%
  \ltx@onelevel@sanitize\CurrentOption
  \fi
  \@ifundefined{KVO@default\@currname.\@currext}{%
\toks\@expandafter{%
  \the\toks\@expandafter,\CurrentOption
}\fi
}}{%
\toks\tw@\expandafter{%
  \the\toks\tw@,\CurrentOption
}\fi
  \toks\@expandafter{%
  \the\toks\@expandafter,\CurrentOption
}}%\fi
\edef\CurrentOption{%\the\toks\tw@%}
@for\CurrentOption:=\CurrentOption\do{%
  \@ifundefined{KV@#1\@KVO@getkey\CurrentOption=\@nil}{}
  \toks\tw@\expandafter{%
    \the\toks\tw@,\CurrentOption
  }%
}%
\toks\@expandafter{%
  \the\toks\@expandafter,\CurrentOption
}%
\fi
\edef\KVO@temp{%endgroup
\noexpand\KVO@calldefault{%the\toks\tw@}%
}%
\edef\KVO@temp{%endgroup
\noexpand\KVO@calldefault{%the\toks\tw@}%
}%
```
Some cleanup of \ProcessOptions.
\let\CurrentOption\@empty
\AtEndOfPackage{\let\@unprocessedoptions\relax}

6.5.3 \ProcessLocalKeyvalOptions

If the optional star is given, we get the family name and expand it for safety.
\newcommand*{\ProcessLocalKeyvalOptions}{%
\if@star{
\begingroup
\edef\x{%
\noexpand\KVO@ProcessLocalKeyvalOptions{\KVO@family}%
}\x}
\KVO@ProcessLocalKeyvalOptions%
}
\def\KVO@ProcessLocalKeyvalOptions#1{%
\let\@tempc\relax
\let\KVO@temp\@empty
Check if \ProcessLocalKeyvalOptions is called inside a package.
\ifx@currext@pkgextension
\else
\PackageError{kvoptions}{\string\ProcessLocalKeyvalOptions space is intended for packages only}%
\fi
The package options are put into toks register \toks@.
\begingroup
\toks\tw@{}%
\ifundefined{opt@\@currname.@currext}{%
\toks\expandafter{%\KVO@family}%
}\{%
\toks\expandafter{\expandafter{\csname opt@\@currname.@currext\endcsname}}%
Without default action we pass all options to \setkeys. Otherwise we have to
check which options are known. These are passed to \setkeys. For the others
the default action is performed.
\ifundefined{\KVO@default@\@currname.@currext}{%
\toks\expandafter{%\KVO@family}%
\toks\expandafter{\expandafter{\expandafter{\KVO@family}}%}
\toks\expandafter{%\csname opt@\@currname.@currext\endcsname}%
\}{%}
\edef\CurrentOption{%\the\toks@%
\toks\expandafter{\expandafter{\expandafter{\KVO@family}}%}
\toks\expandafter{%\csname opt@\@currname.@currext\endcsname}%
\}@for\CurrentOption:=\CurrentOption\do{%
\ifundefined{KV@#1@\expandafter\KVO@getkey\CurrentOption=\@nil}{%
KV@#1\expandafter\KVO@getkey\CurrentOption=\@nil}{
Some cleanup of \ProcessOptions.
\let\CurrentOption@empty
\AtEndOfPackage{\let@unprocessedoptions\relax}%
}

6.5.4 Helper macros

\KVOgetkey Extract the key part of a key=value pair.
\def\KVOgetkey#1=#2\@nil{#1}

\KVOcalldefault
\def\KVOcalldefault#1{%
  \begingroup
  \def\x{#1}%
  \expandafter\endgroup
  \ifx\x\@empty
    \else
      \@for\CurrentOption:=#1\do{%
        \ifx\CurrentOption\@empty
          \else
            \expandafter\KVOsetcurrents\CurrentOption=\@nil
            \@nameuse{KVO@default@\@currname.@\currext}%
          \fi
        \fi
      \fi
  \}
}

\KVOsetcurrents Extract the key part of a key=value pair.
\def\KVOsetcurrents#1=#2\@nil{%
  \def\CurrentOptionValue{#2}%
  \ifx\CurrentOptionValue\@empty
    \let\CurrentOptionKey\CurrentOption
    \let\CurrentOptionValue\relax
  \else
    \edef\CurrentOptionKey{\zap@space#1 \@empty}%
    \expandafter\KVOsetcurrentvalue\CurrentOption\@nil
  \fi
}

\KVOsetcurrentvalue Here the value part is parsed. Package keyval’s \KV@sp@def helps in removing spaces at the begin and end of the value.
6.6 plain TeX
Disable \LaTeX{} stuff.

\begin{verbatim}
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname documentclass\endcsname\relax
\def\ProcessKeyvalOptions{\@ifstar{}\@gobble}
\fi
\KVO@AtEnd
⟨/package⟩
\end{verbatim}

6.7 Package kvoptions-patch

\begin{verbatim}
\NeedsTeXFormat{LaTeX2e}
\begingroup\catcode61\catcode48\catcode32=10\relax
\catcode13=5 % ^^M
\endlinechar=13 %
\catcode123=1 % {
\catcode125=2 % }
\catcode64=11 % @
\def\x{\endgroup
\expandafter\edef\csname KVO@AtEnd\endcsname{\endlinechar=\the\endlinechar\relax
\catcode13=\the\catcode13\relax
\catcode32=\the\catcode32\relax
\catcode35=\the\catcode35\relax
\catcode61=\the\catcode61\relax
\catcode64=\the\catcode64\relax
\catcode123=\the\catcode123\relax
\catcode125=\the\catcode125\relax
}\x
\catcode61\catcode48\catcode32=10\relax
\catcode13=5 % ^^M
\endlinechar=13 %
\catcode35=6 % #
\catcode64=11 % @
\catcode123=1 % {
\catcode125=2 % }
\def\\TMP@EnsureCode#1#2{\edef\KVO@AtEnd{\KVO@AtEnd
\catcode#1=\the\catcode#1\relax
\catcode#1=#2\relax}
\TMP@EnsureCode{39}{12}% '
\TMP@EnsureCode{40}{12}% (
\TMP@EnsureCode{41}{12}% )
\TMP@EnsureCode{43}{12}% +
\TMP@EnsureCode{44}{12}% ,
\end{verbatim}

32
\ProvidesPackage{kvoptions-patch} [%
[2019/11/29 v3.13 LaTeX patch for keyval options (HO)]

Check for \texttt{e-\TeX}.
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname eTeXversion\endcsname\relax
\PackageWarningNoLine{kvoptions-patch}{%
Package loading is aborted, because \texttt{e-\TeX} is missing%
}%
\expandafter\KVO@AtEnd
\fi%
\expandafter\expandafter\expandafter\endgroup
\expandafter\ifetex@unexpanded
\else
\PackageError{kvoptions-patch}{%
Could not find e\TeX's \texttt{\string\unexpanded}.\MessageBreak
Try adding \texttt{\string\RequirePackage\string{etexcmds\string}} %
before \texttt{\string\documentclass}\@ehd
%}
\expandafter\KVO@AtEnd
\fi%
\expandafter\expandafter\expandafter\endgroup
\expandafter\ifpackageloaded{\texttt{xkvltxp}}{%
\PackageWarningNoLine{kvoptions}{%
Option \texttt{\string\patch} cannot be used together with\MessageBreak
package \texttt{\string\xkvltxp} that is already loaded.\MessageBreak
Therefore package loading is aborted%
}%
\KVODatEnd
%}
\def\@if@ptions#1#2#3{%
\begingroup
\KVO@normalize\KVO@temp{#3}%
\edef\x{\endgroup
\noexpand\@if@pti@ns{%
\detokenize\expandafter\expandafter\expandafter{\csname opt@#2.#1\endcsname}
}%
\detokenize\expandafter{\KVO@temp}%
}%
\x%
\def\@pass@ptions#1#2#3{%
\@if@ptions\expandafter\expandafter\expandafter\endgroup
\expandafter\if\csname opt@#2.#1\endcsname
%
\def\ProcessOptions{\let\ds@\@empty \ifundefined{opt@\@currname.\@currext}{\let\@curroptions\@empty}{\expandafter\expandafter\expandafter\def \expandafter\expandafter\expandafter\@curroptions \expandafter\expandafter\expandafter{\csname opt@\@currname.\@currext\endcsname}}\@ifstar\KVO@xprocess@ptions\KVO@process@ptions}
\def\KVO@xprocess@ptions{\ifx\@currext\@clsextension\else\KVO@GetClassOptionsList \@for\CurrentOption:=\KVO@classoptionslist\do{\ifx\CurrentOption\@empty\else\begingroup\ifx\@currext\@clsextension\toks@{}\else\KVO@GetClassOptionsList \toks@\expandafter{\KVO@classoptionslist,}\fi\toks@\tw@\expandafter{\@curroptions}\edef\x{\endgroup\noexpand\in@{,\CurrentOption,}{,\the	oks@\the	oks@\tw@,}}\x\ifin@\KVO@use@ption\edef\ds@\CurrentOption\@empty\fi\fi}}\KVO@process@ptions}
\def\KVO@process@ptions{\ifx\@currext\@clsextension\else\KVO@GetClassOptionsList \@for\CurrentOption:=\@declaredoptions\do{\ifx\CurrentOption\@empty\else\begingroup\ifx\@currext\@clsextension\toks@{}\else\KVO@GetClassOptionsList \toks@\expandafter{\@empty}\fi\toks@\tw@\expandafter{\@curroptions}\edef\x{\endgroup\noexpand\in@{,\CurrentOption,}{,\the	oks@\the	oks@\tw@,}}\x\ifin@\KVO@use@ption\edef\ds@\CurrentOption\@empty\fi\fi}}}
Variant of \ExecuteOptions that better protects \CurrentOption. 
\def\CurrentOption@SaveLevel{0}
\def\ExecuteOptions{%
\expandafter\KVO@ExecuteOptions
\csname CurrentOption\CurrentOption@SaveLevel\endcsname
}
\def\KVODExecuteOptions#1#2{%
\let\CurrentOption\CurrentOption
\def\CurrentOption@SaveLevel{%
\the\numexpr\CurrentOption@SaveLevel+1%
}%
\for\CurrentOption:=#2\do{%
\csname ds@\CurrentOption\endcsname
}%
\edef\CurrentOption@SaveLevel{%
\the\numexpr\CurrentOption@SaveLevel-1%
}%
\let\CurrentOption#1%
}
\def\KVODfileswithoptions#1#2#3#4{%
\ifx#1\@clsextension
\ifx\@classoptionslist\relax
\KVO@normalize\KVO@temp{#2}%
\expandafter\gdef\expandafter\@classoptionslist\expandafter{\KVO@temp}%
\def\reserved@a{\KVO@onefilewithoptions{#3}[{#2}][{#4}]#1%}
%@documentclasshook
\else
\def\reserved@a{\KVO@onefilewithoptions{#3}[{#2}][{#4}]#1%}
\fi
\expandafter\reserved@a
\else
\def\reserved@a{\KVO@onefilewithoptions{#3}[{#2}][{#4}]#1%}
\fi
\fi
\else
\begingroup
\let\KVODtemp\relax
\let\KVODonefilewithoptions\relax
\let@pkgextension\relax
\def@reserved@b##1,{
\ifx\@nil##1\relax
\else
\ifx\relax##1\relax
\else
\KVO@onefilewithoptions{##1}[{\KVO@temp}][{#4}]%
%@pkgextension
\fi
\expandafter\reserved@b
\fi
\fi
\else
\begingroup
\let\KVODtemp\relax
\let\KVODonefilewithoptions\relax
\let@pkgextension\relax
\def@reserved@b##1,{
\ifx\@nil##1\relax
\else
\ifx\relax##1\relax
\else
\KVO@onefilewithoptions{##1}[{\KVO@temp}][{#4}]%
%@pkgextension
\fi
\expandafter\reserved@b
\fi
\fi
\fi

\@undefined
\@unprocessedoptions
\if\@currext\@clsextension\let\LoadClass\@twoloadclasserror\fi
\@popfilename
\@reset@ptions
\reserved@a
\reserved@b
\reserved@c
\reserved@d
\reserved@e
\reserved@a
\reserved@f
\reserved@g
\reserved@h
\reserved@i
\reserved@j
\reserved@k
\reserved@l
\reserved@m
\reserved@n
\reserved@o
\reserved@p
\reserved@q
\reserved@r
\reserved@s
\reserved@t
\reserved@u
\reserved@v
\reserved@w
\reserved@x
\reserved@y
\reserved@z
\reserved@{ }
\reserved@_!
\reserved@!
\reserved@#
\reserved@$
\reserved@$!
\reserved@%
\reserved@&%
\reserved@(*
\reserved@+
\reserved@$+
\reserved@-
\reserved@-
\reserved@0
\reserved@0
\reserved@1
\reserved@1
\reserved@2
\reserved@2
\reserved@3
\reserved@3
\reserved@4
\reserved@4
\reserved@5
\reserved@5
\reserved@6
\reserved@6
\reserved@7
\reserved@7
\reserved@8
\reserved@8
\reserved@9
\reserved@9
\reserved@;?
\reserved@;?
\reserved=@?
\reserved=[?
\reserved=\
\reserved=\
\reserved=\[
\reserved=\[
\reserved={}
\reserved={}
\reserved={}
\reserved={}
\reserved={}:
\reserved={}:
\reserved=;
\reserved=;
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
\reserved=,
7 Test

7.1 Preface for standard catcode check

7.2 Catcode checks for loading
\expandafter\ifx\csname count@\endcsname\relax
\countdef\count@=255 \% 
\fi
\expandafter\ifx\csname @gobble\endcsname\relax
\long\def\@gobble#1{}\%
\fi
\expandafter\ifx\csname @firstofone\endcsname\relax
\long\def\@firstofone#1{#1}\%
\fi
\expandafter\ifx\csname loop\endcsname\relax
\else
\expandafter\@gobble
\fi
{%
\def\loop#1\repeat{% 
\def\body{#1} 
\iterate 
\def\iterate{% 
\body 
\let\next\iterate 
\else 
\let\next\relax 
\fi 
\next 
}%
\let\repeat=\fi 
}%
\def\RestoreCatcodes{}
\count@=0 \% 
\loop 
\edef\RestoreCatcodes{% 
\RestoreCatcodes 
\catcode\the\count@=\the\catcode\count@\relax 
}%
\ifnum\count@<255 \% 
\advance\count@ 1 \% 
\repeat 
\def\RangeCatcodeInvalid#1#2{% 
\count@=#1\relax 
\loop 
\catcode\count@=15 \% 
\ifnum\count@<#2\relax 
\advance\count@ 1 \% 
\repeat 
}%
\def\RangeCatcodeCheck#1#2#3{% 
\count@=#1\relax 
\loop 
\ifnum#3=\catcode\count@ 
\else 
\errmessage{ Character \the\count@\space with wrong catcode \the\catcode\count@\space instead of \number#3} 
\}%
\} 
41
\def\space{ }
\ifnum\count@<#2\relax
\advance\count@ 1 \%
\repeat
\def\LoadCommand{%
\RangeCatcodeInvalid{0}{47}%
\RangeCatcodeInvalid{58}{64}%
\RangeCatcodeInvalid{91}{96}%
\RangeCatcodeInvalid{123}{255}%
\catcode`\@=12 %
\catcode`\=0 %
\catcode`\%=14 %
\LoadCommand
\RangeCatcodeCheck{0}{36}{15}%
\RangeCatcodeCheck{37}{37}{14}%
\RangeCatcodeCheck{38}{47}{15}%
\RangeCatcodeCheck{48}{57}{12}%
\RangeCatcodeCheck{58}{63}{15}%
\RangeCatcodeCheck{64}{64}{12}%
\RangeCatcodeCheck{65}{90}{11}%
\RangeCatcodeCheck{91}{91}{15}%
\RangeCatcodeCheck{92}{92}{0}%
\RangeCatcodeCheck{93}{96}{15}%
\RangeCatcodeCheck{97}{122}{11}%
\RangeCatcodeCheck{123}{255}{15}%
\RestoreCatcodes
}\Test
\csname @@end\endcsname
\end
(<test1)
(*test2)
\NeedsTeXFormat{LaTeX2e}
\makeatletter
\catcode`\@=11 %
\def\RestoreCatcodes{}
\count@=0 %
\loop
  \edef\RestoreCatcodes{%
    \RestoreCatcodes
    \catcode\the\count@=\the\catcode\count@\relax
  }%
  \ifnum\count@<255 %
  \advance\count@\@ne
  \repeat
\def\RangeCatcodeInvalid#1#2{%
  \count@=#1\relax
  \loop
    \catcode\count@=15 %
    \ifnum\count@<#2\relax
    \advance\count@\@ne
    \repeat
  \end}%
\edef\RestoreCatcodes{%
  \RestoreCatcodes
  \catcode\the\count@=\the\catcode\count@\relax
}%
\def\Test#1{%  
\RangeCatcodeInvalid{0}{47}%
\RangeCatcodeInvalid{58}{64}%
\RangeCatcodeInvalid{91}{96}%
\RangeCatcodeInvalid{123}{255}%
\catcode'@=12 %
\catcode'\=0 %
\catcode'\{=1 %
\catcode'\}=2 %
\catcode'#=6 %
\catcode'\[=12 %
\catcode'\]=12 %
\catcode'%=14 %
\catcode'\ =10 %
\catcode13=5 %
#1\relax
\RestoreCatcodes%
}\Test{\RequirePackage{kvoptions-patch}}%
\Test{\RequirePackage{kvoptions}}%
\csname @@end\endcsname
⟨/test2⟩
⟨*/test3⟩
\NeedsTeXFormat{LaTeX2e}
\makeatletter
\RequirePackage{kvoptions}[2019/11/29]
\def\msg#1{\immediate\write16{#1}}
\define@key{testfamily}{testkey}{\msg{[testfamily/testkey/#1]}}
\define@key{testfamily}{testdefaultkey}{\msg{[testfamily/testdefaultkey/#1]}}
\AddToKeyvalOption{testfamily}{testkey}{\msg{[addition/#1]}}
\AddToKeyvalOption{testfamily}{testdefaultkey}{\msg{[addition/#1]}}
\setkeys{testfamily}{testkey=testA, testdefaultkey=testB, testdefaultkey,}
\SetupKeyvalOptions{family=testfamily}
\AddToKeyvalOption{testkey}{\msg{[star addition/#1]}}
\AddToKeyvalOption{testdefaultkey}{\msg{[star addition/#1]}}
\setkeys{testfamily}{testkey=testA, testdefaultkey=testB, testdefaultkey,}
8 Installation

8.1 Download

Package. This package is available on CTAN:\(^1\):


Bundle. All the packages of the bundle ‘kvoptions’ 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/kvoptions.tds.zip

TDS refers to the standard “A Directory Structure for \TeX Files” (CTAN:pkg/tds). Directories with \texttt{texmf} in their name are usually organized this way.

\(^1\)CTAN:pkg/kvoptions
8.2 Bundle installation

Unpacking. Unpack the kvoptions.tds.zip in the TDS tree (also known as \texttt{texmf} tree) of your choice. Example (linux):

\begin{verbatim}
unzip kvoptions.tds.zip -d ~/texmf
\end{verbatim}

8.3 Package installation

Unpacking. The \texttt{.dtx} file is a self-extracting \texttt{docstrip} archive. The files are extracted by running the \texttt{.dtx} through plain \TeX:

\begin{verbatim}
tex kvoptions.dtx
\end{verbatim}

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

\begin{verbatim}
kvoptions.sty \rightarrow tex/latex/kvoptions/kvoptions.sty
kvoptions-patch.sty \rightarrow tex/latex/kvoptions/kvoptions-patch.sty
kvoptions.pdf \rightarrow doc/latex/kvoptions/kvoptions.pdf
example-mycolorsetup.sty \rightarrow doc/latex/kvoptions/example-mycolorsetup.sty
kvoptions.dtx \rightarrow source/latex/kvoptions/kvoptions.dtx
\end{verbatim}

If you have a \texttt{docstrip.cfg} that configures and enables \texttt{docstrip}'s TDS installing feature, then some files can already be in the right place, see the documentation of \texttt{docstrip}.

8.4 Refresh file name databases

If your \TeX{} distribution (\TeX{} Live, mikTeX, …) relies on file name databases, you must refresh these. For example, \TeX{} Live users run \texttt{texhash} or \texttt{mktexlsr}.

8.5 Some details for the interested

Unpacking with \LaTeX\. The \texttt{.dtx} chooses its action depending on the format:

\texttt{plain \TeX}: Run \texttt{docstrip} and extract the files.

\texttt{\LaTeX}: Generate the documentation.

If you insist on using \LaTeX\ for \texttt{docstrip} (really, \texttt{docstrip} does not need \LaTeX), then inform the autodetect routine about your intention:

\begin{verbatim}
latex \let\install=y\input{kvoptions.dtx}
\end{verbatim}

Do not forget to quote the argument according to the demands of your shell.

Generating the documentation. You can use both the \texttt{.dtx} or the \texttt{.drv} to generate the documentation. The process can be configured by the configuration file \texttt{ltxdoc.cfg}. For instance, put this line into this file, if you want to have A4 as paper format:

\begin{verbatim}
\PassOptionsToClass{a4paper}{article}
\end{verbatim}

An example follows how to generate the documentation with pdf\LaTeX:

\begin{verbatim}
pdflatex kvoptions.dtx
makeindex -s gind.ist kvoptions.idx
pdflatex kvoptions.dtx
makeindex -s gind.ist kvoptions.idx
pdflatex kvoptions.dtx
\end{verbatim}
9 References


[8] Package tracefnt, Frank Mittelbach, Rainer Schöpf, 1997/05/29. CTAN:pkg/tracefnt

[9] Package xkeyval, Hendri Adriaens, 2005/05/07. CTAN:pkg/xkeyval

[10] The L\TeX3 Project, \LaTeX\ 2ε for class and package writers, 2003/12/09. CTAN:pkg/clsguide

10 History

[0000/00/00 v0.0]
- Probably David Carlisle’s code in hyperref was the start.

[2004/02/22 v1.0]
- The first version was never published. It also has offered a patch to get rid of \LaTeX\’s option expansion.

[2006/02/16 v2.0]
- Now the package is redesigned with an easier user interface.
- \ProcessKeyvalOptions remains the central service, inherited from hyperref’s \ProcessOptionsWithKV. Now the use inside classes is also supported.
- Provides help macros for boolean and simple string options.
- Fixes for the patch of \LaTeX. The patch is only enabled, if the user requests it.
[2006/02/20 v2.1]
- Unused option list is sanitized to prevent problems with other packages that use own processing methods for key value options. Disadvantage: the unused global option detection is weakened.
- New option type by `\DeclareVoidOption` for options without value.
- Default rule by `\DeclareDefaultOption`.
- Dynamic options: `\DisableKeyvalOption`.

[2006/06/01 v2.2]
- Fixes for option patch.

[2006/08/17 v2.3]
- `\DeclareBooleanOption` renamed to `\DeclareBoolOption` to avoid a name clash with package `\ifoption`.

[2006/08/22 v2.4]
- Option patch: `\ExecuteOptions` does not change the meaning of macro `\CurrentOption` at all.

[2007/04/11 v2.5]
- Line ends sanitized.

[2007/05/06 v2.6]
- Uses package `etexcmds`.

[2007/06/11 v2.7]
- The patch part fixes LaTeX bug latex/3965.

[2007/10/02 v2.8]
- Compatibility for plain TeX added.
- Typos in documentation fixed (Axel Sommerfeldt).

[2007/10/11 v2.9]
- Bug fix for option patch.

[2007/10/18 v3.0]
- New package `kvoptions-patch`.

[2009/04/10 v3.1]
- Space by line end removed in definition of internal macro.
• \ProcessLocalKeyvalOptions added.
• \DisableKeyvalOption with the action=ignore option fixed (Joseph Wright).

• \DeclareLocalOption, \DeclareLocalOptions added.

• Documentation addition: recommendation for Joseph Wright’s review article.
• Documentation addition: local/global options.

• \AddToKeyvalOption added.

• Fix: If a default handler is configured, it is now also called for classes.

• Missing space in error message added.

• Documentation for package kvoptions-patch improved. No code changes.

• Key setkeys added for \SetupKeyvalOptions.

• \DeclareVoidOption also parses the second parameter as \TeX argument to improve compatibility with \DeclareOption.

• Fix because of design bug in package xkeyval that removes global options with equal signs.

• Documentation updates.

• Documentation updates.
11 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; plain numbers refer to the code lines where the entry is used.

Symbols

\# ................................. 1371, 1499
\% ................................. 1447, 1502
\$ ................................. 1372, 1445, 1470, 1495
\@end ................................ 1548
\@filesin@ptions ................. 1079, 1357
\@Unprocessedoptions ........... 1210, 1245
\@SetupDriver .................... 41, 43
\@badrequireerror ............... 1358
\@classoptionslist .............. 658, 659, 1125, 1127
\@cls@pkg ......................... 1186, 1221, 1236, 1239
\@clsextension ................... 1228
\@current ......................... 231, 242, 245, 248, 253, 256,

\@curname ......................... 57, 228, 242, 245, 246,
248, 253, 256, 257, 259, 264,
267, 270, 294, 316, 334, 411,
433, 479, 728, 732, 741, 767,
819, 823, 825, 866, 1000, 1006,
1016, 1036, 1095, 1166, 1167,
1172, 1173, 1176, 1181, 1203,
1205, 1207, 1209, 1211, 1212,
1216, 1223, 1227, 1246, 1247, 1251
\@currname ......................... 1001,
1004, 1022, 1068, 1248, 1250, 1253
\@declaredoptions ............... 1012, 1042, 1075
\@documentclasshook ............. 1132
\@ehc .............................. 322, 513, 562, 598, 815
\@ehd .............................. 958
\@empty ............................ 88, 89, 495, 544, 687,
745, 795, 810, 852, 860, 863,
873, 877, 999, 1001, 1013, 1029,
1040, 1045, 1078, 1097, 1155,
1167, 1168, 1205, 1248, 1254,
1262, 1282, 1283, 1296, 1339, 1344
\@expandtwoargs .................. 719
\@filesin@ptions ................ 1079, 1358, 1360
\@firstofone ..................... 355, 358, 426, 1380, 1383
\@firstoftwo ..................... 1274
\@for ................................ 693,
737, 774, 832, 862, 1012, 1039,
1055, 1068, 1075, 1115, 1253, 1338
\@gobble ......................... 369, 540, 627, 887, 1377, 1385
\@@gobblewo ..................... 424
\@@ifOpt@ns ...................... 973
\@@ifOptions ...................... 969, 1173
\@@ifdefinable ................... 349
\@@if@aded ......................... 1172
\@@if@ter ......................... 1216
\@@ifnextchar .................... 387
\@@ifpackageloaded .............. 961
\@@ifstar ......................... 612, 676, 799, 887, 1009
\@@ifndef ......................... 424
\@@latex@error ................... 1186, 1234
\@@latex@warning@no@line ........ 1218
\@@let@token ...................... 1298, 1302
\@@missingfileerror ............. 1209
\@@namedef ....................... 395, 479
\@@nameuse ....................... 866
\@@ne .............................. 233, 1479, 1487
\@@nil ............................. 695, 698, 739,
776, 834, 855, 865, 871, 878,
881, 1145, 1156, 1263, 1266,
1268, 1270, 1279, 1298, 1305,
1310, 1312, 1315, 1317, 1321, 1334
\@@pass@ptions ................... 983, 1203
\@@pkgextension .................. 298,
338, 437, 502, 811, 1143, 1150, 1246
\@@popfilename ................... 1228
\@@pushfilename .................. 1164
\@@removelement ................. 719, 1085
\@@reset@ptions .................. 1169, 1229
\@@secondoftwo .................. 1276
\@@tempc .......................... 686, 809
\@@twoloadclasserror ............ 1227
\@@typeset@protect .............. 225
\@@undefined ...................... 155, 1213
\@@unknownoptionerror .......... 68, 1233, 1254
\@@Unprocessedoptions ............ 796, 853, 1080, 1210, 1214
\@@unusedoptionlist ............. 720, 745, 746, 749,
751, 1088, 1091, 1096, 1097, 1099
\@@x@protect ..................... 222
\[ ................................ 1500
\[ ................................ 397, 1273, 1446, 1496
\{ ................................ 1369, 1497
\} ................................ 1370, 1498
\] ................................ 1501