The l3keys2e package
LaTeX 2ε option processing using LaTeX3 keys[*]

The LaTeX3 Project[†]

Released 2013/03/12

The key–value method for optional arguments is very popular, as it allows the class or package author to define a large number of options with a simple interface. The expl3 bundle of LaTeX3 base code includes the module l3keys for defining keys, but to use these when loading LaTeX 2ε packages and classes requires extra support. That support is provided by this small package, which is intended to enable LaTeX 2ε packages to benefit from the power of the LaTeX3 key–value system.

0.1 Creating and using keyval options

As with any key–value input, using key–value pairs as package or class options has two parts. The first stage is to define one or more keys, using the `\keys_define:nn` function. For example, an option which simply stores a value would be created using:

```
\keys_define:nn { module }
  { option .set:N = \l_module_variable_tl }
```

On its own, this will not make the key an option for the package or class containing the definition. The second stage is therefore to process the current options, searching for applicable keys.

`\ProcessKeysOptions`

`\ProcessKeysOptions {⟨module⟩}`

The `\ProcessKeysOptions` function is used to check the current option list against the keys defined for `{`⟨*module*⟩`}`. Global (class) options and local (package) options are checked when this function is called in a package. Each option which does match a key name is then used to attempt to set the appropriate key using `\keys_set:nn`. For example, the option defined earlier would be processed by the line

```
\ProcessKeysOptions { module }
```

[*] This file describes v4467, last revised 2013/03/12.

[†] E-mail: latex-team@latex-project.org

`\ProcessKeysPackageOptions`

`\ProcessKeysPackageOptions {⟨module⟩}`

This function works in a similar manner to `\ProcessKeysOptions`. When used in a LaTeX 2$_\varepsilon$ package, `\ProcessKeysPackageOptions` will not examine any class options available. In contrast, `\ProcessKeysOptions` does parse class options (in common with the LaTeX 2$_\varepsilon$ kernel function `\ProcessOptions`).

0.2 l3keys2e Implementation

```
⟨*package⟩
⟨@@=keys⟩
\ProvidesExplPackage
  {\ExplFileName}{\ExplFileDate}{\ExplFileVersion}{\ExplFileDescription}
```

`\l__keys_latexe_options_clist`

A single list is used for all options, into which they are collected before processing.

```
\clist_new:N \l__keys_latexe_options_clist
```

(*End definition for* `\l__keys_latexe_options_clist`. *This function is documented on page* **??**.)

`\l__keys_process_class_bool`

A flag to indicate that class options should be processed for packages.

```
\bool_new:N \l__keys_process_class_bool
```

(*End definition for* `\l__keys_process_class_bool`. *This function is documented on page* **??**.)

`\__keys_latexe_options:n`

The main function calls functions to collect up the global and local options into `\l__keys_latexe_options_clist` before calling the underlying functions to actually do the processing. So that a suitable message is produced if the option is unknown, the special `unknown` key is set if it does not already exist for the current module.

```
\cs_new_protected:Npn \__keys_latexe_options:n #1
  {
    \clist_clear:N \l__keys_latexe_options_clist
    \__keys_latexe_options_global:n {#1}
    \__keys_latexe_options_local:
    \keys_if_exist:nnF {#1} { unknown }
      {
        \keys_define:nn {#1}
          {
            unknown .code:n =
              {
                \msg_error:nnxx { keyvalue } { option-unknown }
                  { \l_keys_key_tl } { \@currname }
              }
          }
      }
    \keys_set:nV {#1} \l__keys_latexe_options_clist
    \AtEndOfPackage { \cs_set_eq:NN \@unprocessedoptions \scan_stop: }
  }
```

(*End definition for* `\__keys_latexe_options:n`. *This function is documented on page* **??**.)

__keys_latexe_options_global:n

Global (class) options are handled differently for LaTeX 2ε packages and classes. Hence this function is essentially a check on the current file type. The initial test is needed as LaTeX 2ε allows variables to be equal to \scan_stop:, which is forbidden in LaTeX3 code.

```
\cs_new_protected:Npn \__keys_latexe_options_global:n #1
  {
    \cs_if_eq:NNF \@classoptionslist \scan_stop:
      {
        \cs_if_eq:NNTF \@currext \@clsextension
          { \__keys_latexe_options_class:n {#1} }
          {
            \bool_if:NT \l__keys_process_class_bool
             { \__keys_latexe_options_package:n {#1} }
          }
      }
  }
```

(*End definition for* __keys_latexe_options_global:n*. This function is documented on page* **??***.*)

__keys_latexe_options_class:n

For classes, each option (stripped of any content after =) is checked for existence as a key. If found, the option is added to the combined list for processing. On the other hand, unused options are stored up in \@unusedoptionlist. Before any of that, though, there is a simple check to see if there is an unknown key. If there is, then *everything* will match and the mapping can be skipped.

```
\cs_new_protected:Npn \__keys_latexe_options_class:n #1
  {
    \keys_if_exist:nnTF {#1} { unknown }
      { \clist_put_right:No \l__keys_latexe_options_clist \@classoptionslist }
      {
        \clist_map_inline:Nn \@classoptionslist
          {
            \keys_if_exist:nnTF {#1} { \__keys_latexe_remove_equals:n {##1} }
              { \clist_put_right:Nn \l__keys_latexe_options_clist {##1} }
              { \clist_put_right:Nn \@unusedoptionlist {##1} }
          }
      }
  }
```

(*End definition for* __keys_latexe_options_class:n*. This function is documented on page* **??***.*)

__keys_latexe_options_package:n

For global options when processing a package, the tasks are slightly different from those for a class. The check is the same, but here there is nothing to do if the option is not applicable. Each valid option also needs to be removed from \@unusedoptionlist.

```
\cs_new_protected:Npn \__keys_latexe_options_package:n #1
  {
    \clist_map_inline:Nn \@classoptionslist
      {
        \keys_if_exist:nnT {#1} { \__keys_latexe_remove_equals:n {##1} }
          {
            \clist_put_right:Nn \l__keys_latexe_options_clist {##1}
            \clist_remove_all:Nn \@unusedoptionlist {##1}
          }
      }
  }
```

(*End definition for* `\__keys_latexe_options_package:n`*. This function is documented on page* ??.)

__keys_latexe_options_local: If local options are found, the are added to the processing list. LaTeX 2ε stores options for each file in a macro which may or may not exist, hence the need to use `\cs_if_exist:c`.

```
\cs_new_protected_nopar:Npn \__keys_latexe_options_local:
  {
    \cs_if_eq:NNF \@currext \@clsextension
      {
        \cs_if_exist:cT { opt@ \@currname . \@currext }
          {
            \exp_args:NNc \clist_put_right:NV \l__keys_latexe_options_clist
              { opt@ \@currname . \@currext }
          }
      }
  }
```

(*End definition for* `\__keys_latexe_options_local:`*. This function is documented on page* ??.)

__keys_latexe_remove_equals:n
__keys_latexe_remove_equals:w As the name suggests, this is a simple function to remove an equals sign from the input. This is all wrapped up in an `n` function so that there will always be a sign available.

```
\cs_new:Npn \__keys_latexe_remove_equals:n #1
  { \__keys_latexe_remove_equals:w #1 = \q_stop }
\cs_new:Npn \__keys_latexe_remove_equals:w #1 = #2 \q_stop {#1}
```

(*End definition for* `\__keys_latexe_remove_equals:n`*. This function is documented on page* ??.)

\ProcessKeysOptions
\ProcessKeysOptions The user macro are simply wrappers around the internal process. In contrast to other similar packages, the module name is always required here.

```
\cs_new_protected_nopar:Npn \ProcessKeysOptions #1
  {
    \bool_set_true:N \l__keys_process_class_bool
    \__keys_latexe_options:n {#1}
  }
\cs_new_protected_nopar:Npn \ProcessKeysPackageOptions #1
  {
    \bool_set_false:N \l__keys_process_class_bool
    \__keys_latexe_options:n {#1}
  }
\@onlypreamble \ProcessKeysOptions
\@onlypreamble \ProcessKeysPackageOptions
```

(*End definition for* `\ProcessKeysOptions`*. This function is documented on page* 1.)

One message to give.

```
\msg_new:nnnn { keyvalue } { option-unknown }
  { Unknown~option~'#1'~for~package~#2. }
  {
    LaTeX~has~been~asked~to~set~an~option~called~'#1'~
    but~the~#2~package~has~not~created~an~option~with~this~name.
  }
```

⟨/package⟩

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.