TEX and Copyediting

CV Radhakrishnan*, CV Rajagopal† and SK Venkatesan‡

Released 2014/07/26

Copyedit implements copyediting support for LATEX documents. Authors can enjoy the freedom of using, for example, using words with US or UK or Canadian or Australian spelling in a mixed way, yet, they can choose any one of the usage form for their entire document irrespective of kinds of spelling they have adopted. In the same fashion, the users can have the benefit of following features available in the package:

1. Localization — British-American-Australian-Canadian
2. Close-up, Hyphenation, and Spaced words
3. Latin abbreviations
4. Acronyms and Abbreviations
5. Itemization, nonlocal lists and labels
6. Parenthetical and serial commas
7. Non-local tokenization in language through Abbreviations and pronouns.

1 Package Loading

The copyedit package can be loaded with the command:

```
\usepackage[<options>]{copyedit}
```

\ceset
\cesetup

```
\ceset {⟨options⟩}
\cesetup {⟨options⟩}
```

There is a user-friendly function, namely, \ceset or \cesetup available which can be used anywhere in the document to change the options or behaviour of the package from that point onwards. The usage of the function is:

The ⟨options⟩ shall be provided as a comma separated list.

*JWRA 34, Jagathy, Trivandrum 695014, India, E-mail: <cvr@cvr.cc>
†SRA 34c, Elipode, Trivandrum 695013, India, E-mail: <cvr3@cvr.cc>
‡TNQ Books and Journals, Kottivakkam, Chennai 600041, India, E-mail: <skvenkat@tnq.co.in>

2 Using the Features

2.1 Localization — British, American, Australian and Canadian

\definevariant `\definevariant{⟨uk⟩}{⟨us⟩}{⟨au⟩}{⟨ca⟩}`

The package provides a database of commonly used English language words in all the above four variant spelling. One can add new entries to the list with the command: The variants, ⟨*uk*⟩, ⟨*us*⟩, ⟨*au*⟩ and ⟨*ca*⟩ denote British, American, Australian and Canadian spelling respectively.

\vara `\vara {⟨variant⟩}`

In the document, one can use the macro `\vara{⟨variant⟩}`. The user is free to use a combination of words of various spelling, but output will totally be dependent on the option of locale chosen for the document. For the sake of distinction and to prove our point, let us assume that the word ⟨*color*⟩ has the following four different spelling in four different locale:

1. UK: colour
2. US: color
3. AU: COLOUR
4. CA: COLOR

By using the four words in the following manner:

```
\vara{colour} \vara{color} \vara{COLOUR} \vara{COLOR}
```

and by using various language option with different values for the option `{⟨lang⟩}`, we can switch across different languages for all four words, no matter, whatever locate spelling one has adopted for the word in the document instance.

1. `lang=uk` ⟶ colour colour colour colour
2. `lang=us` ⟶ color color color color
3. ...

starred \vara `\vara*{⟨word⟩}`

If for any reason, the user chooses to retain the word as it is then it is possible by using the starred version of the macro:

2.2 Close-up, Hyphenation, Spaced Words

\hyp	\hyp{⟨word⟩}{⟨word⟩}
\closeup	\closeup{⟨word⟩}{⟨word⟩}
\sword	\sword{⟨word⟩}{⟨word⟩}

There are three commands that make the above possible:

1. \hyp{anti}{body} ⟶ anti-body (hyphenate)
2. \closeup{anti}{body} ⟶ antibody (close up)
3. \sword{Civil}{War} ⟶ Civil War (space our)

2.3 Latin Abbreviation

\definelat	\definelat {⟨abbr⟩} {⟨abbr without dots⟩} {⟨English form⟩}
\lat	\lat {⟨abbr⟩}

Latin abbreviations can be used in different ways. The option lat=0 will keep the Latin abbreviation as such in form in italic shape. abbr=italic will make the abbreviation italicized (which is also default). abbr=rm will make it upshape. lat=1 will take away the periods in the abbreviated Latin forms while lat=2 will replace the abbreviation with its English language equivalents.

The package comes with a default list of abbreviations. Users shall use the command \lat{⟨abbr⟩} to invoke the desired format of abbeviation in the document instance.

2.4 Acronyms

\newacro	\newacro {⟨short form⟩} {⟨long form⟩}
\ac	\ac {⟨short form⟩}

The highly aclaimed package, namely, acronym has been made use of to achieve the extensive features available therein. Kindly refer to its documentation for getting an understanding of the usage.

2.5 Itemization and Non-local Lists

A list environment, namely, eitem has been defined which can be used for a variety of purposes as enlisted below by changing its options.

Option	Rendering
eitem=0	normal list: Firstly, Secondly, Thirdly, ...
eitem=1	normal list: First, Second, Third, ...
eitem=2	enumerated list: 1, 2, 3, ...
eitem=3	itemized bulletted list
eitem=4	para list: Firstly, secondly, thirdly, ...
eitem=5	para list: First, second, third, ...

In the para list forms, each item will be separated by semicolon (;) and the penultimate item will be connected to the last one by the 'and' automatically.

\elist `\elist {⟨comma separated list⟩}`

There is also a convenient command, "elist" that helps to format a comma separated list in a proper way with proper spacing. For example,

```
\elist{warblers,tits,wrens,hummingbirds}
\elist{warblers , tits, wrens ,hummingbirds}
```

will be formatted in the proper way with right spacing and correct punctuation:

warblers, tits, wrens and hummingbirds

2.6 Parenthetical and Serial Comma

\pc `\pc {⟨text to be included⟩}`

This is accomplished with the command \pc and the text to be distiguished shall be its argument. Different values for option with the same name, pc, provide different functionality as detailed below:

Option	Rendering
pc=0	argument is separated by parenthetical commas
pc=1	separated by parenthesis
pc=2	separated by emdashes
pc=3	formatted as a footnote
pc=4	formatted as a sidenote (marginpar)

2.7 Non-local Tokenization

\definetoken `\definetoken {⟨label⟩} {⟨full tokens⟩} {⟨less tokens⟩}`
\tkn `{⟨least tokens⟩}`
`\tkn {⟨label⟩}`

A sequence of minimization operation can be brought out by first defining the tokens to be minimized and then using the same consecutively will typeset the tokens in differently and minimized manner each time it is called. We define a token as:

```
\definetoken{mango}{His Holyness, the Prince of Mangoistan}
                   {The Prince of Mangoistan}{He}
```

will have different output as given below:

Command	**Rendering**
First instance: `\tkn{mango}`	His Holyness, the Prince of Mangoistan
Second: `\tkn{mango}`	The Prince of Mangoistan
Third: `\tkn{mango}`	He
Fourth: `\tkn{mango}`	He

3 copyedit implementation

```
⟨*initex | package⟩
⟨@@=cedt⟩
\ProvidesExplPackage
  {\ExplFileName}{\ExplFileDate}{\ExplFileVersion}{\ExplFileDescription}
```

`\__cedt_load_check:n`

There are also a number of packages that are incompatible with copyedit. These are all checked for next. Some of the incompatible packages will not raise an error if loaded after copyedit. So a test is made at the beginning of the document as well. The message for this may be needed immediately, so it is created here not with the other messages.

```
\msg_new:nnnn { copyedit } { incompatible-package }
  { Package~'#1'~incompatible. }
  { The~#1~package~and~copyedit~are~incompatible. }
\cs_new_protected:Npn \__cedt_load_check:n #1
  {
    \group_begin:
    \@ifpackageloaded {#1}
    { \msg_error:nnx { copyedit } { incompatible-package } {#1} }
    { }
    \group_end:
  }
\clist_map_function:nN
  { Array , MyPackage }
  \__cedt_load_check:n
\AtBeginDocument {
  \clist_map_function:nN { Array , MyPackage }
    \__cedt_load_check:n
  }
```

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

`\wrAux`

```
\NewDocumentCommand \wrAux { m }
   { \iow_now:Nx \@auxout { #1 } }
```

A simple scratch macro to write out stuff to the auxiliary file. There should be some elegant way or macro for the job.
(*End definition for* `\wrAux`. *This function is documented on page* **??**.)

`\l__cedt_lat_int` `\l__cedt_pc_int` `\l__cedt_lang_int` `\l__cedt_eitem_int`

Needed counters and functions to use the counters are defined in advance.

```
\int_new:N \l__cedt_lat_int
\NewDocumentCommand \setlat { m }
  { \int_set:Nn \l__cedt_lat_int { #1 } }

\int_new:N \l__cedt_pc_int
\NewDocumentCommand \setpc { m }
  { \int_set:Nn \l__cedt_pc_int { #1 } }

\int_new:N \l__cedt_lang_int
\NewDocumentCommand \setlang { m }
  { \int_set:Nn \l__cedt_lang_int { #1 } }

\int_new:N \l__cedt_eitem_int
\NewDocumentCommand \seteitem { m }
  { \int_set:Nn \l__cedt_eitem_int { #1 } }
\int_new:N \l__cedt_abbr_int

\NewDocumentCommand \setabbr { m }
  {
    \str_if_eq:nnTF { #1 } { italic }
      { \int_set:Nn \l__cedt_abbr_int { 0 } }
      { \int_set:Nn \l__cedt_abbr_int { 1 } }
  }
```

(*End definition for* `\l__cedt_lat_int` *and others. These functions are documented on page* **??**.)

`\___cedt_lang_check:n` Define a function to check language code and set the language numeric counter.

```
\cs_new_protected:Npn \___cedt_lang_check:n #1
  {
    \str_if_eq:nnT { #1 } { uk } { \setlang { 0 } }
    \str_if_eq:nnT { #1 } { us } { \setlang { 1 } }
    \str_if_eq:nnT { #1 } { ca } { \setlang { 2 } }
    \str_if_eq:nnT { #1 } { au } { \setlang { 3 } }
  }
```

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

Define key-values as per expl3 syntax:

```
\keys_define:nn { copyedit }
  {
    lang        .code:n     = \___cedt_lang_check:n { #1 } ,
    lat         .code:n     = \setlat { #1 }              ,
    abbr        .code:n     = \setabbr{ #1 }              ,
    pc          .code:n     = \setpc { #1 }               ,
    draft       .bool_set:N = \l__cedt_draft_bool          ,
    last        .bool_set:N = \l__cedt_last_bool           ,
    eitem       .code:n     = \seteitem { #1 }            ,
```

```
    key-unknown .code:n     =
    {
      \msg_error:nnx { copyedit } { unknown-option }
       { \exp_not:V \l_keys_key_tl }
    }
  }
```

Set key-values and process key-value options.

```
\keys_set:nn { copyedit }
  {
    lang      = { uk }     ,
    lat       = { 0 }      ,
    abbr      = { italic } ,
    pc        = { 0 }      ,
    draft     = { false }  ,
    last      = { false }  ,
    eitem     = { 0 }      ,
  }

\ProcessKeysOptions { copyedit }
```

(*End definition for . These functions are documented on page* **??**.)

`\ceset` `\cesetup`

A macro, `\ceset` has been defined to invoke any option at any point in the document instance. A variant, `\cesetup`, has also been defined.

```
\NewDocumentCommand \ceset { m }
  {
    \keys_set:nn { copyedit } { #1 }
    \___cedt_lang_check:n { \l__cedt_lang_tl }
  }
\cs_set_eq:NN \cesetup \ceset
```

(*End definition for* `\ceset` *and* `\cesetup`*. These functions are documented on page* 1.)

`\switchvariant` `\definevariant` `\vara`

`\swtchvariant` is an internal function that will help to switch between different locale as per the language option chosen. `\definevariant` is the one for defining different locale and `\vara` is the user level command for using in document instance.

```
\NewDocumentCommand \switchvariant { m m m m }
   {
     \int_case:nn { \l__cedt_lang_int }
     {
       { 0 } { #1 }
       { 1 } { #2 }
       { 2 } { #3 }
       { 3 } { #4 }
     }
   }

\NewDocumentCommand \definevariant { m m m m }
   {
```

```
    \tl_set:cn { g_vara_#1_tl }
        { \switchvariant { #1 } { #2 } { #3 } { #4 } }
    \tl_set:cn { g_vara_#2_tl }
        { \switchvariant { #1 } { #2 } { #3 } { #4 } }
    \tl_set:cn { g_vara_#3_tl }
        { \switchvariant { #1 } { #2 } { #3 } { #4 } }
    \tl_set:cn { g_vara_#4_tl }
        { \switchvariant { #1 } { #2 } { #3 } { #4 } }
  }

\DeclareDocumentCommand \vara { s m }
  {
    \IfBooleanTF {#1}
    { #2 }
    { \normalvara {#2} }
  }

\NewDocumentCommand \normalvara { m }
  {
    \use:c { g_vara_#1_tl }
  }
```

(*End definition for* `\switchvariant`, `\definevariant`, *and* `\vara`. *These functions are documented on page* 2.)

`\hyp` `\closeup` `sword`

`\hyp`, `\closeup` and `\sword` are three simple macros to hyphenate, close up and separate two words respectively.

```
\NewDocumentCommand \hyp { m m } { #1-#2 }
\NewDocumentCommand \closeup { m m } { #1#2 }
\NewDocumentCommand \sword { m m } { #1~#2 }
```

(*End definition for* `\hyp`, `\closeup`, *and* `sword`. *These functions are documented on page* 3.)

`\definelat` `\lat`

Latin abbreviation and its variant forms can be brought in by these macros.

```
\NewDocumentCommand \definelat { m m m }
  {
    \tl_set:cn { g__cedt_lat_#1_tl }
    {
      \group_begin:
      \int_case:nn { \l__cedt_abbr_int}
      {
        { 0 } { \itshape }
        { 1 } { \upshape }
      }
      \int_case:nn { \l__cedt_lat_int }
      {
        { 0 } { #1 }
        { 1 } { #2 }
        { 2 } { #3 }
      }
      \group_end:
    }
```

```
  }

\NewDocumentCommand \lat { m } { \use:c { g__cedt_lat_#1_tl } }
```

(*End definition for* `\definelat` *and* `\lat`*. These functions are documented on page 3.*)

`\pc` `\elist` Parenthetical comma (pc) and its variants are chosen with this macro in combination with different options.

```
\NewDocumentCommand \pc { m }
  {
    \int_case:nn { \l__cedt_pc_int }
    {
      { 0 } { \unskip,~#1,~ }
      { 1 } { (#1) }
      { 2 } { ---~#1~--- }
      { 3 } { \unskip\footnote{#1} }
      { 4 } { \marginpar{ \footnotesize\raggedright#1} }
    }
  }
\clist_new:N \l__cedt_clist
\NewDocumentCommand \elist { m }
  {
    \clist_set:Nn \l__cedt_clist { #1 }
    \clist_use:Nnnn \l__cedt_clist { ~and~ } { ,~ } { ~and~ }
  }
```

(*End definition for* `\pc` *and* `\elist`*. These functions are documented on page 4.*)

`\definetoken` `\tkn` `\Token` Non-local tokenization

```
\NewDocumentCommand \definetoken { m m m m }
  {
    \tl_gset:cn { #1num } { 0 }
    \tl_gset:cn { g_toks_#1_0_tl } { #2 \tl_gset:cn { #1num } { 1 } }
    \tl_gset:cn { g_toks_#1_1_tl } { #3 \tl_gset:cn { #1num } { 2 } }
    \tl_gset:cn { g_toks_#1_2_tl } { #4 \tl_gset:cn { #1num } { 2 } }
  }
\NewDocumentCommand \tkn { m }
  { \use:c { g_toks_#1_\use:c{#1num}_tl } }
\cs_set_eq:NN \Token \tkn
```

(*End definition for* `\definetoken`*,* `\tkn`*, and* `\Token`*. These functions are documented on page* **??**.)

`eitem` `eitem` is defined to switch between non-local list of different types. We make use of the package enumitem for this purpose.

```
\RequirePackage{enumitem}
\chardef\thre@=3
\newenvironment{enumerate*}[1][]%
     {\@nameuse{enit@enumerate*}\enitdp@enumerate{enum}\thre@{#1}}
     {\@nameuse{enit@endenumerate*}}
```

A few macros like, \lastlabel, \mysep, \myseplast, \elistcnt, \LastItem are defined to make the job easier.

```
\int_new:N \l__cedt_elistcnt_int
\NewDocumentCommand \lastlabel { } { \tex_xdef:D\@itemlabel{Lastly}}
\NewDocumentCommand \mysep { } { ;\hskip .5em plus .1em minus .1em  }
\NewDocumentCommand \myseplast { } {\space and \space  }
\NewDocumentCommand \elistcnt { } { \int_use:N \l__cedt_elistcnt_int }
\NewDocumentCommand \LastItem { m m }
   { \tl_gset:cn { l__cedt_tmpa_#1_tl } { #2 } }
```

\checklast

\checklast is the macro that finds the last item number and substitutes with Last or Lastly depending upon the option chosen.

```
\NewDocumentCommand \checklast { m }
  {
    \tl_if_exist:cTF { l__cedt_tmpa_ \elistcnt _tl }
      { \int_set:Nn \l_tmpa_int { \use:c{ l__cedt_tmpa_\elistcnt _tl }} }
      { \int_set:Nn \l_tmpa_int { 0 } }
    \int_set:Nn \l_tmpb_int { \the\c@enumi }
    \int_compare:nNnTF { \l_tmpa_int } = { \l_tmpb_int }
        {
          \bool_if:NTF \l__cedt_last_bool
          {
            \int_case:nn { \l__cedt_eitem_int }
            {
              { 0 } { Lastly, }
              { 1 } { Last, }
              { 4 } { lastly, }
              { 5 } { last, }
            }
          }
          { #1 }
        }
        { #1 }
  }
```

Smallish set up changes are made to the enumitem to suit the requirements.

\ctext
\@ctext
\ltext
\@ltext
\paratext
\@paractext
\paraltext
\@paraltext

These macros act as variables to hold textual values for each item number.

```
\makeatletter
\def\ctext#1{\expandafter\@ctext\csname c@#1\endcsname}
\def\@ctext#1{\ifcase#1\or \checklast{First,}\or
  \checklast{Second,}\or \checklast{Third,}\or
  \checklast{Fourth,}\or \checklast{Fifth,}\or
  \checklast{Sixth,}\or \checklast{Seventh,}\or
  \checklast{Eigth,}\or \checklast{Nineth,}\or
  \checklast{Tenth,}\fi}
\AddEnumerateCounter{\ctext}{\@ctext}{Second,}

\def\ltext#1{\expandafter\@ltext\csname c@#1\endcsname}
```

```
\def\@ltext#1{\ifcase#1\or \checklast{Firstly,}\or
  \checklast{Secondly,}\or \checklast{Thirdly,}\or
  \checklast{Fourthly,}\or \checklast{Fifthly,}\or
  \checklast{Sixthly,}\or \checklast{Seventhly,}\or
  \checklast{Eigthly,}\or \checklast{Ninethly,}\or
  \checklast{Tenthly,}\fi}
\AddEnumerateCounter{\ltext}{\@ltext}{Secondly,}

\def\paractext#1{\expandafter\@paractext\csname c@#1\endcsname}
\def\@paractext#1{\ifcase#1\or \checklast{First,}\or
  \checklast{second,}\or \checklast{third,}\or
  \checklast{fourth,}\or \checklast{fifth,}\or
  \checklast{sixth,}\or \checklast{seventh,}\or
  \checklast{eigth,}\or \checklast{nineth,}\or
  \checklast{tenth,}\fi}
\AddEnumerateCounter{\paractext}{\@paractext}{second,}

\def\paraltext#1{\expandafter\@paraltext\csname c@#1\endcsname}
\def\@paraltext#1{\ifcase#1\or \checklast{Firstly,}\or
  \checklast{secondly,}\or \checklast{thirdly,}\or
  \checklast{fourthly,}\or \checklast{fifthly,}\or
  \checklast{sixthly,}\or \checklast{seventhly,}\or
  \checklast{eigthly,}\or \checklast{ninethly,}\or
  \checklast{tenthly,}\fi}
\AddEnumerateCounter{\paraltext}{\@paraltext}{secondly,}
```

At last `eitem` has been defined.

```
\NewDocumentEnvironment { eitem } { }
   {
     \int_gincr:N \l__cedt_elistcnt_int
     \int_case:nn { \l__cedt_eitem_int }
     {
       { 0 } { \begin{enumerate}
               [label=\ltext*,align=left,itemjoin=\mysep,itemjoin*=\myseplast] }
       { 1 } { \begin{enumerate}
               [label=\ctext*,align=left,itemjoin=\mysep,itemjoin*=\myseplast] }
       { 2 } { \begin{enumerate}[label=\arabic*.] }
       { 3 } { \begin{enumerate}[label=\textbullet] }
       { 4 } { \begin{enumerate*}
               [label=\paraltext*,itemjoin=\mysep,itemjoin*=\myseplast] }
       { 5 } { \begin{enumerate*}
               [label=\paractext*,itemjoin=\mysep,itemjoin*=\myseplast] }
     }
    }
    {
      \wrAux { \token_to_str:N \LastItem
        { \int_use:N \l__cedt_elistcnt_int } {\the\c@enumi} }
      \int_case:nn { \l__cedt_eitem_int }
     {
       { 0 } { \end{enumerate} }
```

```
        { 1 } { \end{enumerate} }
        { 2 } { \end{enumerate} }
        { 3 } { \end{enumerate} }
        { 4 } { \end{enumerate*} }
        { 5 } { \end{enumerate*} }
      }
    }
%
```

⟨/initex | 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.

U

V

W