The cjw-ltr Class*

Colin J. Wynne†

1997/01/01

Contents

Introduction

I recently made the change to LaTeX 2ε after some years of dedication to Plain TeX and that code which I had already written for it. Among that code were macros for writing letters. Since I am studying in Germany this year, I had also written a set of macros for writing proper letters according to accepted German format.

After switching to LaTeX, I of course gave the letter class a try. I didn't like some of the default behaviour that the class produces. Furthermore, there was no support for writing German letters. I am aware that a style file dinbrief exists for LaTeX 2.09 or LaTeX 2ε in compatibility mode, but I would prefer to use something exclusively LaTeX 2ε. Since I planned on doing some cosmetic alterations to the basic letter class anyway, I figured it wouldn't be too much extra to include the capability of formatting German letters as well.

So, this is what I have come up with. Feel free to e-mail me comments, suggestions, bugs, etc.

1 Creating Letters

`\letterskip` The first problem that I had with the letter class was spacing. I do not like having the paragraphs of my letters unindented and separated by white space. This poses a problem since the vertical space between the components of the letter are defined (nay, *hard-wired*) as multiples of `\parskip`. Therefore, all vertical space has been defined in terms of `\letterskip`, a length which is set initially to 0.7 em, which happens to be the letter class's default value for `\parskip`.

*This file has version 0.9 as of 1997/01/01.

†E-Mail at: cwynne@mts.jhu.edu, cwynne@jhu.edu .

1.1 Standard Letters

letter
fax

American format letters are still delineated by the letter environment. No changes have been made here; the single mandatory argument (which may be left empty) is still the address of the recipient. I have, however, added a macro `\fax`, analagous to `\telephone`, for including a fax number in the return address. If the recipient's fax is also desired, it should probably just be included in the argument to the letter environment, perhaps using the `\faxlabel` macro, as well (see below).

Now, one of the things that I have always been taught about proper business letters is that the closing and the sender's address should line up, and be as far right as possible. The standard letter class does not do this; instead, the address is set as far right as possible and the closing begins `\indentedwidth` from the right margin. By default, this is exactly in the middle of the page. I can not express satisfactorily just how ugly and unbalanced this seems to me. So, first I have `\let` the length `\indentedwidth` to the slightly more intuitive `\addresswidth`. When the letter's `\opening` is declared, `\addresswidth` is fixed at the size of the return address. Both the sending address and the closing will be set at in a box of width `\addresswidth`; the box is flush right, the material within the box is flush left.

\addresswidth

\opening

The `\opening` macro still takes the text of the opening as its one argument, and still begins the letter by typesetting the addresses and date. There are a few differences in the typesetting, though. First, if a `\telephone` command has been entered, the number will be set as the last line of the address. Then comes the skip between the address and the date. This skip is rather smaller that that provided by the letter class. I found this to be preferable. Then come the recipient's address and the greeting. After these are typeset, the `\parindent` is set to 20 pt and the `\parskip` to 0 pt plus 1 pt. The values can be reset after the `\opening` call and before the text of the body if such is desired.

\closing

The only change to the closing macro is the substitution of `\letterskip` for the original `\medskipamount`. This means that all vertical space in a letter—between address and date and recipient's address, between address and greeting, etc.—are proportional. I considered defining each of those quantities to be their own macro, such as `\beforedateskip`, `\beforerecipientskip`, and so forth, but that seemed a little bit excessive. If, however, such flexibility is desired, feel free to ask for it.

1.2 German Letters

brief

*The environment for a German letter is, appropriately enough, named brief (the German word for letter). As for `begin{letter}`, the invocation takes the recipient's address as the one mandatory argument. However, in standard format, the address line containing the *Postleitzahl* (zip code, postal code—whatever) and the city is separated by a bit of whitespace from the rest of the address. Thus, whereas other lines of the address are delineated by the `\\` macro, the sequence `!!` indicates the beginning of the PLZ. Thus, my former address would most likely be entered as:

```
\begin{brief}{%
```

*Dies soll wirklich auf Deutsch sein, oder? Das hatte ich betrachtet, aber es schien mir ein bißchen durcheinander, die beiden Sprachen in dieser Dokumentation zu mischen. Wenn jemand die Übersetzung will, könnte er mich fragen. Anderseits ist jeder frei, die Übersetzung selbst zu erledigen.

```
Herrn                \\
Colin J. Wynne       \\
Klosterweg 28 / L110 !!
76131 Karlsruhe}
```

\totitle
\toPLZ

I have not included any macros for creating mailing labels—other than the standard macros from the letter class. However, if someone feels the need to write such, then the following is significant. For a standard letter, the first line is taken as the recipient's 'title'. This can be a proper academic or professional title, or, more commonly, simply a polite *Herrn*, as in the example above. This is stored in the macro `\totitle`. The PLZ information (everything in the argument after `!!`) is stored in `\toPLZ`. These complement the standard `\toname` and `\toaddress`. I would add appropriate macros myself, but I have no facilities for printing mailing labels to check the result. I imagine, though, that setting

```
\edef\toname{\totitle \\ \toname}
\edef\toaddress{\toaddress \\ \toPLZ}
```

would allow existing labelling functions to suffice. (`\edef` to avoid a recursive definition, of course.)

\ort
\datum
\heute
\absender
\telefon
\unterschrift

German equivalents are allowed for the standard information providing macros. `\ort` is equivalent to `\location`, and can be used to provide the place name on the dateline. The date itself can be entered with the `\datum` command, and the `\heute` macro produces the current date in German format. Thus, the input `\ort{Karlsruhe} \datum{\heute}` could produce the dateline 'Karlsruhe, den 14. Mai 1995' at the top of the letter. `\absender` is used as `\address` to indicate the sender's address, with `\telefon` to provide a telephone number, and `\fax` is of course the same as in English. Finally, `\unterschrift` provides the signature after the closing. Commands are also given for entering the optional *Betreff* and *Bezug* information. A note to purists: one should include both or neither of these lines. By default, the labels "Betreff" and "Bezug" are printed at the beginning of their respective lines. The actual label (including an empty label) can be entered as an optional argument to the `\betreff` or `\bezug` commands.

\betreff
\bezug

\anrede
\gruss

The opening and closing of a letter are called by macros named for the German equivalents to `\opening` and `\closing`, namely `\anrede` and `\gruss`.

\anlage

Enclosures are indicated by the `\anlage` command, which calls the standard `\encl` macro. `\anlage` takes an optional argument, namely the `\enclname` macro, which is used to label the enclosures. By default, "Anlage" is the label. Since multiple enclosures are fairly standard, one can enter simply 'n' (that is, `\anlage[n]`⟨*enclosures*⟩) and produce the expected label "Anlagen."

2 The Code

The very first thing to do is to include the file `letter.cls`, since most of what's in there is very good, and I don't want to have to rewrite it and bloat this file. We make sure as well that it inherits any options.

```
\DeclareOption*{\PassOptionsToClass{\CurrentOption}{letter}}
\ProcessOptions

\LoadClass{letter}
```

Now we can start to change things. I think the default \headsep is too large; so, we will make it a bit smaller.

```
\setlength\headsep   {25\p@}
```

Next, we provide the length that will be used for vertical space within a letter. We initialise it to a reasonable value.

```
\newlength{\letterskip}
  \setlength{\letterskip}{0.7em}
```

2.1 Standard Letters

\addresswidth We first declare the length \addresswidth, which is used to determine how far from the right margin the sender's adress and the closing should be set. Since the letter class uses the \indentedwidth length to help place the sender's address, we will simply recycle that length for our own purposes.

```
\let\addresswidth\indentedwidth
```

\faxlabel \tellabel If a fax number is included, it should be labeled so as not to be confused with a telephone number. The labels (appropriate to both German and English) will simply be 'Tel:' and 'Fax:'. But, in the interest of æsthetics, we want the printed versions to have the same width, so that the actual numbers line up nicely. Since 'Fax:' is clearly the wider label, we use it to determine the width we want.

```
\newlength\lblwd
{\setbox0=\hbox{Fax:\enskip}\global\setlength{\lblwd}{\wd0}}
\newcommand*{\faxlabel}{\hbox to\lblwd{Fax:\enskip\hfil}}
\newcommand*{\tellabel}{\hbox to\lblwd{Tel:\enskip\hfil}}
```

\fax The \fax command simply sets a global placeholder.

\check@iffax

```
\newcommand*{\fax}[1]{\def\faxnum{#1}}
  \fax{}
```

The \check@iffax macro inserts the labels if appropriate.

```
\def\check@iffax{%
  \ifx\@empty\faxnum\else
    \edef\faxnum{\protect\faxlabel\faxnum}
    \ifx\@empty\telephonenum\else
      \edef\telephonenum{\protect\tellabel\telephonenum}
  \fi\fi}
```

\opening The \opening macro will collect all information for the letter environment and begin the typesetting. First, we calculate the actual width of the given address.

```
\def\opening#1{%
```

The first step is to determine what actual address information we have. We will figure this out and put it into a placeholder. If a fax number is included, we will want to use labels for telephone and fax.

```
 \check@iffax
 \def\from@address{%
```

We see if a return address was supplied. If not, we simply set the date.

```
    \ifx\@empty\fromaddress%
      \@date%
```

If so, we typeset the address including a telephone number if one is supplied. Then comes a bit of whitespace and the date.

```
    \else% home address
      \fromaddress%
      \ifx\@empty\telephonenum\else%
        \\ \telephonenum%
      \fi%
      \ifx\@empty\faxnum\else%
        \\ \faxnum%
      \fi%
      \\*[\letterskip]\@date%
    \fi}
  \setbox0=\hbox{%
    \begin{tabular}{l}\from@address\end{tabular}}
```

With these values we set out global value for `\addresswidth`.

```
  \global\setlength{\addresswidth}{\wd0}
```

With no address, we want the `firstpage` pagestyle.

```
  \ifx\@empty\fromaddress
    \thispagestyle{firstpage}%
```

With one, we want the `empty` pagestyle for the first page.

```
  \else
    \thispagestyle{empty}
  \fi
```

In either case, we fill horizontally and dump the box with the address information.

```
  \noindent\hfill\box0
  \par
```

After the date comes some more whitespace and the recipient—

```
  \vspace{2\letterskip}%
  {\raggedright\toname\\ \toaddress\par}%
```

—then more whitespace and the text of the greeting. Note that all of the whitespace is given in terms of `\letterskip`.

```
  \vspace{2\letterskip}%
  \noindent#1%
```

Finally, we reset the values of `\parindent` and `\parskip` and inhibit a break before the body. (What set of addresses would make TEX *want* to break after the greeting?)

```
  \setlength{\parindent}{20\p@}
  \setlength{\parskip}{0\p@ \@plus\@ne\p@\relax}
  \par\nobreak}
```

`\closing` The `\closing` macro is unchanged from the letter class, except for the change from `\parindent` to `\letterskip` and the use of `\addresswidth` in determining the box size.

```
\long\def\closing#1{\par\nobreak\vspace{2\letterskip}%
  \stopbreaks
  \noindent\hfill
  \parbox{\addresswidth}{\raggedright%
       #1\\[6\letterskip]%
       \ifx\@empty\fromsig
```

```
            \fromname
        \else
    \fromsig
        \fi\strut}
    \par}
```

2.2 German Letters

First, we `\let` several appropriate macros for German usage.

```
\let\unterschrift \signature
\let\absender     \address
\let\ort          \location
\let\telefon      \telephone
\let\datum        \date
```

`\heute` Here are some simple definitions which I use in my own macro files.

```
\newcommand{\theday}{\number\day\relax}
\newcommand{\themonth}{%
  \ifcase\month\or January\or February\or%
  March\or April\or May\or June\or July\or August\or%
  September\or October\or November\or December\fi}
\newcommand{\themonat}{%
  \ifcase\month\or Januar\or Februar\or%
  M\"arz\or April\or Mai\or Juni\or Juli\or August\or%
  September\or Oktober\or November\or Dezember\fi}
\newcommand{\theyear}{\number\year\relax}
```

From these we can construct a German date macro, `\heute`.

```
\newcommand{\heute}{den~\theday.\ \themonat\ \theyear}
```

These are also useful if one prefers a different (*i.e.*, date first) format for the `\today` macro.

`\betreff` `\bezug` Next come the macros for the additional information which may be included in the headers of a German letter.

```
\newcommand{\betreff}[2][Betreff]{%
  \def\betreffname{#1}\long\def\@betreff{#2}}
\newcommand{\bezug}[2][Bezug]{%
  \def\bezugname{#1}\long\def\@bezug{#2}}
```

These are initialised to be empty.

```
\betreff[]{}
\bezug[]{}
```

`brief` Now comes the brief environment itself. German letters are suppose to use unindented and separated paragraphs, so we will set the proper values as soon as the environment is entered.

```
\long\def\brief#1{%
  \newpage
  \setlength{\parindent}{0pt}
  \setlength{\parskip}{0.7em}
```

Further initialisations are taken from the `\letter` macro.

```
    \if@twoside \ifodd\c@page
                \else\thispagestyle{empty} \hbox{}\newpage\fi
```

```
    \fi
    \c@page\@ne
    \interlinepenalty=200 % Smaller than the TeXbook value
```

Since the supplied argument has more information in it than that given to a letter environment, we must have our own equivalent to the letter class's `\@processto`.

```
    \@prozessan{\leavevmode\ignorespaces #1}
```

Lastly, we make the default date a German one.

```
    \date{\heute}}
```

`\@prozessan` The address processing is based on the `\@processto` macro from letter.cls. We call the first subprocess, `\@xproz`, to fetch the PLZ part of the address.

```
\long\def\@prozessan#1{%
  \@xproz #1!!@@@%
```

If there is a PLZ, the first test includes an extra occurrence of `!!` in the `\toPLZ` macro. So, we call `\@xproz` again without the extra characters (which were necesary to make sure that we didn't get a 'usage doesn't match definition' error the first time around).

```
  \ifx\@empty\toPLZ\else\@xproz #1@@@\fi%
```

The `\@xproz` macro has now stored all the address parts before the PLZ in `\@tempa`. Now there are up to three different parts that may be in `\@tempa` which we will try to separate out. We use `\expandafter` in order that the tokens within `\@tempa` get properly considered.

```
  \expandafter\@yproz \@tempa\\@@@%
```

The `\@yproz` splits the input into `\@tempb`, the part before the first linebreak, and `\@tempc`, the rest. If the second part, that in `\@tempc`, is empty, then we want to set the third part, `\@tempd`, to be empty, too.

```
  \ifx\@empty\@tempc
    \let\@tempd\@empty
```

Otherwise, we call `\@zproz` to further subdivide the address input.

```
  \else
    \expandafter\@zproz \@tempa\\@@@%
```

If there is a third part, `\@tempd`, we have the same problem with extra characters as above in `\toPLZ`. We call `\@zproz` one more time in order to read `\@tempd` correctly.

```
    \ifx\@empty\@tempd
      \else
      \expandafter\@zproz \@tempa{}@@@%
    \fi
  \fi
```

Now we need to decide what to call whichever address pieces actually exist. Obviously, if everything back to `\@tempb` was empty, all of the address parts should be likewise.

```
  \ifx\@empty\@tempb
    \let\totitle\@empty \let\toname\@empty \let\toaddress\@empty
  \else
```

Now we have at least one address part. If that is the only one, it is most certainly not a title, so `\totitle` must be set empty.

```
    \ifx\@empty\@tempc
      \let\totitle\@empty
```

One line with a PLZ is almost assuredly an address proper. Without a PLZ it is probably a name.

```
      \ifx\@empty\toPLZ
        \let\toname\@tempb \let\toaddress\@empty
      \else
        \let\toname\@empty \let\toaddress\@tempb
      \fi
```

If `\@tempb` and `\@tempc` contain something, but `\@tempd` is empty, the two lines should be name and address.

```
    \else
      \ifx\@empty\@tempd
        \let\totitle\@empty \let\toname\@tempb \let\toaddress\@tempc
```

The last option is that title, name and address are all present.

```
      \else
        \let\totitle\@tempb \let\toname\@tempc \let\toaddress\@tempd
      \fi
    \fi
    \fi}
```

This is my second implementation of the `\@prozessan` macro. The first was truly ugly and had the added disadvantage of, well, not working. In fact, it failed miserably for a few relatively simple cases. That the current version (as far I can tell) works does not, however, excuse the fact that it is still somewhat ugly and seems to me to be very efficient. I am still open to any suggested improvements.

The subproceses to `\@prozessan` are simply designed to be able to make assignments based on the structure of the passed argument.

```
\long\def\@xproz #1!!#2@@@{\def\@tempa{#1}\def\toPLZ{#2}}
\long\def\@yproz #1\\#2@@@{\def\@tempb{#1}\def\@tempc{#2}\def\@tempd{}}
\long\def\@zproz #1\\#2\\#3@@@{\def\@tempc{#2}\def\@tempd{#3}}
```

The end of the brief environment is identical to that of the letter environment.

```
\let\endbrief\endletter
```

`\anrede` As for the `\opening` macro in the letter environment, the `\anrede` macro signals the beginning of the letter. We begin by setting the dateline.

```
\def\anrede#1{%
  \check@iffax
  \ifx\@empty\fromlocation
    \def\@dateline{\@date}
  \else
    \def\@dateline{\fromlocation,\space\@date}
  \fi
```

Set with a typewriter, a German letter should leave two blank lines before the return address. Since we are using something a little more capable than a typewriter, we will leave a small amount of whitespace, in the form of a multiple of `\letterskip`.

```
\null\vspace{1.2\letterskip}
\noindent\ignorespaces
```

If there is no return address (a situation which I don't actually see happening with this format), just set the dateline at the right margin.

```
\ifx\@empty\fromaddress%
  \pagestyle{firstpage}%
  {\raggedleft \@dateline}%
```

Otherwise, the address is set at the left margin and the dateline right justified at the level of the first line of the sender information. A telephone number and fax, if provided, follow the address.

```
\else % Home address
  \pagestyle{empty}%
  {\raggedright%
     \fromname \hfill \@dateline\\
     \fromaddress}
      \ifx\@empty\telephonenum
      \else
        \\ \telephonenum
      \fi
      \ifx\@empty\faxnum
      \else
        \\ \faxnum
      \fi
\fi
\par
```

Next comes whitespace approximately twice as large as the topskip.

```
\vspace{2.2\letterskip}
```

We then set the recipient's address. Using the information provided from the `\@prozessan` macro above, we include those pieces which were supplied.

```
{\raggedright
   \ifx\@empty\totitle
   \else
     \totitle \\
   \fi
   \toname
   \ifx\@empty\toaddress
   \else
     \\ \toaddress
   \fi
   \ifx\@empty\toPLZ
   \else
     \\*[0.6\letterskip]\toPLZ
   \fi
 \par}
```

After the address come, if supplied, the *Betreff* and *Bezug* lines. First, though, we check to see if only one is supplied—if so, we issue a warning.

```
\@checkbetreffbezug
```

Now we can typeset those lines. A small amount of whitespace separates them from one another and from the address.

```
   \ifx\@empty\@betreff
```

```
    \else
      \vspace{\letterskip}
      \ifx\@empty\betreffname
      \else
        \betreffname:\quad
      \fi
      \@betreff\par
    \fi
    \ifx\@empty\@bezug
    \else
      \vspace{1\letterskip}
      \ifx\@empty\bezugname
      \else
        \bezugname:\quad
      \fi
      \@bezug\par
    \fi
```

A slightly larger amount of whitespace precedes the actual *Anrede*.

```
  \vspace{1.4\letterskip}%
  #1\par\nobreak}
```

The `\@checkbetreffbezug` macro emits a warning if only one of those two lines is supplied.

```
\def\@checkbetreffbezug{%
  \ifx\@empty\@betreff
    \ifx\@empty\@bezug
    \else
      \ClassWarningNoLine{cjw-ltr}{%
        Die \protect\betreff und \protect\bezug Befehlen sollen nur
                                                    \MessageBreak
        zusammen verwendet werden.  Ihnen fehlt die  \MessageBreak
        Betreff-Zeile.}
    \fi
  \else
    \ifx\@empty\@bezug
      \ClassWarningNoLine{cjw-ltr}{%
        Die \protect\betreff und \protect\bezug Befehlen sollen nur
                                                    \MessageBreak
        zusammen verwendet werden.  Ihnen fehlt die  \MessageBreak
        Bezug-Zeile.}
    \fi
  \fi}
```

`\gruss` The command `\gruss` ends a letter by first adding some whitespace and disabling further pagebreaks.

```
\long\def\gruss#1{\par\nobreak\vspace{2\letterskip}%
  \stopbreaks
  \noindent%
```

The closing itself is set, leaving enough whitespace for the actual signature.

```
  {\raggedright #1 \\[5\letterskip]
     \ifx\@empty\fromsig
       \fromname
```

```
    \else
      \fromsig
    \fi}
\par
```

Finally, a bit of white space separates the bottom of the closing from any postscripts.

```
\vspace{\letterskip}}
```

`\anlage` Enclosures, *Anlagen*, are specified by the `\anlage` command. As with `\betreff` and `\bezug`, the label is given by an optional argument.

```
\newcommand{\anlage}[1][Anlage]{%
```

If the optional argument is simply 'n', we pluralize the label.

```
\def\@tempa{n} \def\@tempb{#1}
\ifx\@tempa\@tempb
  \renewcommand{\enclname}{Anlagen}
```

Otherwise, the optional argument (or the default, if none is given) becomes the label.

```
\else
  \renewcommand{\enclname}{\@tempb}
\fi\encl}
```

And with that, we are done.