# Termcal.sty—printing a class calendar*

Bill Mitchell

January 8, 2014

**Abstract**

This package is intended to print a term calendar for use in planning a class. It has a flexible mechanism for specifying which days of the week are to be included and for inserting text either regularly on the same day each week, or on selected days, or for a series of consecutive days. It also has a flexible mechinism for specifing class and nonclass days. Text may be inserted into consecutive days so that it automatically flows around nonclass days.

## 1 Description

`calendar` The main command in this package is the `calendar` environment. Figure 1 shows an example of its use, which might be suitable for a Monday-Wednesday-Friday class with a Thursday recitation in a mercifully short term. The LaTeX input to generate it is given in figure 2.

Figure 1: Example of the use of the `calendar` environment

Figure 2: LaTeX input for Figure 1

The two arguments to the `calendar` environment are the starting date, in the format `m/d/y`, and the number of weeks to be printed. The contents of

*Version: 1.8. Date: 1997/01/11. Documentation: 1996/01/10

the environment describe what is to appear in the calendar. The commands fall into two classes: those which specify what is to be printed on a particular day in each week, and those which specify what is to be printed on specific days during the term.

The commands which specify what each week is to look like are `\calday` and `\skipday`. These commands specify the days of the week in order; thus there should be seven of them (athough ten may be prefered for a course

`\skipday` covering the French revolution). The macro `\skipday` simply declares that the corresponding day should not be printed in the exam; thus the sample calendar has boxes only for Monday, Wednesday, Thursday, and Friday. The

`\calday` macro `\calday[`*optional head*`]{`*option list*`}` is used to specify a day which is to be printed. The required argument *option list* is a (possibly empty) list of TEX commands which are executed before the text for that day is printed. It can (at least in principal) include any LATEX command, but several commands, described below, are provided specifically for this purpose. The optional argument is the text of a heading for that day which will at the top of each page of the calendar.

`\classday` `\noclassday` **Available options.** The macros `\classday` and `\noclassday` declare that the specified day is, or is not, a class day. Days specified as class days are numbered and can be refered to by their numbers. The command

`\weeklytext` `\weeklytext{`*text*`}` specifies that the indicated text should appear every week in the box for that day. The *text* may include TEX commands; for example the following could be used to number regular Monday quizes:

```
\weeklytext{\stepcounter{quiznumber} Quiz~\arabic{quiznumber}}
```

**Options for a specific day** are specified by the command `\options{`*day*`}{`*option*

`\options` *list*`}`. The *option list* argument is exactly the same as for `\calday`. The options added by `\options` are executed after those for `\calday`, and thus may be used to modify or nullify for a specific day the general instructions specified for a particular day of the week. If there no quiz is planned for Halloween then `\options{10/31/94}{\weeklytext{}}` could be used to supress printing the usual weekly message.

There are two ways of specifing the *day* argument of `\options`. First, the date may be specified, in the format `m/d/y`; for example

```
\options{11/11/94}{\noclass}
```

specifies, that November 11th, Veterans's day, is not a class day. The second method of specifying the date is by its number:

```
\options{C6}{\weeklytext{}}
```

would supress the normal text on the sixth day of class. Of course, only days specified as class days can be addressed in this style.

**Inserting Text.** There is a similar provision for text to be printed on a specific day, using the command `\caltext`. The command

`\caltext`

```
\caltext{10/31/94}{Halloween\\No Quiz!}
```

will print the indicated message on October 31, and `\caltext{C6}{Hour Exam}` will print the indicated message on the sixth class day.

Two extra commands are provided to simplify the job of entering text for consecutive class days, as in specifying the lecture topic or homework for each day. The command `\caltexton{`*class day*`}{`*text*`}` specifies a starting day and inserts the indicated text on that day. The command `\caltextnext{`*text*`}` can then be used to print text on successive class days. The command `\caltextnext{}`, with an empty argument, may be used to skip days.

`\caltexton` `\caltextnext`

## 1.1 Modifying the style of the calendar

Several parameters and commands will allow some modfication of the style of the entire calendar. The size of the calendar is specified by `\calboxdepth`, which specifies the minimum height of the box for each day, and `\calwidth` which specifies the width of the calendar. The defaults are 1 inch for `\calboxdepth` and `\textwidth` for `\calboxwidth`.

`\calboxdepth` `\calwidth`

The printing of the date and classnumber in each box is done by the commands `\calprintdate` and `\calprintclass`. The default definitions of these macros are as follows:

`\calprintdate`

```
\newcommand{\calprintdate}{%
    \ifnewmonth\framebox{\monthname\ \ordinaldate}%
    \else \ordinaldate\fi
 }
```

`\calprintclass`

```
\newcommand{\calprintclass}{\textbf{\small\theclassnum}}
```

They may be changed with \renewcommand.
By default the calendar will automatically be split over several pages. This can be avoided by putting the entire calendar in a \vbox. It will normally also be necessary to change \calboxdepth so that the calendar will fit on one page.

# 2 The Code

```
\ProvidesPackage{termcal}[\filedate\space\fileversion\space
                          Latex2e package to print a Term calendar]
\NeedsTeXFormat{LaTeX2e}
```

We use the package longtable so that the calendar can be split over several pages if desired.

```
\RequirePackage{longtable}
\RequirePackage{ifthen}
```

Parameters determining the size of the calendar.

```
\newlength{\calboxdepth}\setlength\calboxdepth{1in}
\newlength\calwidth\setlength\calwidth{\textwidth}
\newlength{\ca@boxwidth}   %% set by \endcalendar
```

The following parameters are used to control the construction of the calendar.

ca@dpw The number of days used in a week.

```
\newcounter{ca@dpw}
```

ca@numwks The number of weeks in the calendar.

```
\newcounter{ca@numwks} %% weeks in calendar
```

ca@wknum Used to keep track of the current week.

```
\newcounter{ca@wknum}
```

\ca@doaweek Two token boxes used to build up the basic contents of the calendar. \ca@doaweek is built up by \calday, and prints a typical week. It will be filled up by the macros \calday and \skipday.

```
\newtoks\ca@doaweek
```

\ca@doweeks This will be essentially ca@numwks copys of \ca@doaweek.

```
\newtoks\ca@doweeks
```

\ca@colhead
\ifca@chead

The column headings which appear at the top of every page is collected in the token box \ca@colhead by \calday. The switch \ifca@chead is set to true if any such column heads are specified.

```
\newtoks\ca@colhead
\newif\ifca@chead
```

calendar

This is the basic environment. The \calendar command only saves the parameters and initializes some counters.

```
\newenvironment{calendar}[2]%
 {%
   \setcounter{ca@numwks}{#2}
   \setdate{#1}
   \setcounter{ca@dpw}{0}
   \setcounter{classnum}{1}
 }
```

The calendar is actually created in the code for \endcalendar. It will be printed as a longtable. Since the longtable and tabular environments don't work well with loops in their body, we will build up the body in a token box, \ca@doweeks.

\ca@doweeks

```
  {
   \ifca@chead\ca@doweeks{\the\ca@colhead\endhead\hline\hline}\fi
   \setcounter{ca@wknum}{0}
   \whiledo{\value{ca@wknum}<\value{ca@numwks}}%
            {\stepcounter{ca@wknum}%
             \addtotoks{\ca@doweeks}{\the\ca@doaweek\\\hline}}
```

Now we calculate the widths of the boxes, using a formula from the Latex Companion.

```
  \ca@boxwidth=\calwidth
  \divide\ca@boxwidth by \c@ca@dpw\relax
  \advance\ca@boxwidth by -2\tabcolsep\relax
  \setlength\@tempdima\arrayrulewidth\relax
  \multiply\@tempdima\c@ca@dpw\relax
  \advance\@tempdima\arrayrulewidth\relax
  \divide\@tempdima\c@ca@dpw\relax
  \advance\ca@boxwidth by -\@tempdima\relax
```

Now we use the longtable environment to print out the calendar.

```
\begin{longtable}[l]
        {|%
         *{\theca@dpw}{p{\ca@boxwidth}|}%
         @{}}%
   \hline
   \the\ca@doweeks
 \end{longtable}}
```

\addtotoks The first argument is a tokenbox, and the second argument is a list of tokens to be added to the end of its current contents.

```
\newcommand\addtotoks[2]{#1\expandafter{\the#1#2}}
```

\calday Now the commands used to build up a typical week. They work by filling up the token box \ca@doaweek. We also fill up the token box \ca@colhead to give column headings.

```
\ca@doaweek={\stepcounter{ca@wknum}%
            \ignorespaces}
\newcommand\calday[2][]{\stepcounter{ca@dpw}%
        \ifca@fday\addtotoks\ca@doaweek{&}\addtotoks\ca@colhead{&}\fi
       \addtotoks\ca@doaweek{\ca@doaday{#2}}
        \def\@tempa{#1}\ifx\@tempa\@empty
           \else\addtotoks\ca@colhead{\strut\scshape\centering #1}\ca@cheadtrue\fi
        \ca@fdaytrue
}
```

\ifca@fday Is it the first day? This determines whether & needs to be added as a separator.

```
\newif\ifca@fday
```

\skipday

```
\newcommand\skipday{\addtotoks\ca@doaweek{\advancedate}}
```

After a couple of preliminaries, we will define the command \ca@doaday which is actually prints out the text for each day of the calendar.

classnum This is the counter used to keep track of class days. It is initialized to 1 in the beginning of the calendar environment.

```
\newcounter{classnum}
```

\ca@normbs The meaning of \\ is changed by the `longtable` environment. We save its standard meaning so that it can be used in the text to be printed in the calendar boxes.

```
\let\ca@normbs=\\
```

\ca@doaday The command \ca@doaday does the actual printing of the contents of the box for each day. First the options are read, in the following order: options specified in the argument to \calday, then options specified by date, and finally options specified by classday.

```
\newcommand\ca@doaday[1]{
   \hbox{\vrule depth \calboxdepth height 0pt width 0pt\vtop{
   #1%                                  %options specified by |\calday|
   \csname\curdate options\endcsname%  % options specified by date
   \ifclassday\csname C\theclassnum options\endcsname\fi  %by classnumber
```

Then the heading is printed.

```
   \hbox to \hsize{\calprintdate\hfill\ifclassday\calprintclass\fi}
   \vspace{2pt}
```

Now we are ready to print the text. We do it inside a group in which the normal meaning of \\ is restored.

```
      \begingroup
        \let\\=\ca@normbs
        \raggedright
        \sloppy

        \the\weeklytext\par
        \csname\curdate text\endcsname
        \ifclassday\csname C\theclassnum text\endcsname
                   \stepcounter{classnum}\fi
      \endgroup
   }}      % end of hbox containing the days calendar text.
```

Finally we advance the date. The command \advancedate will set \newmonthtrue if appropriate.

```
    \global\newmonthfalse
    \advancedate
}
```

## 2.1 Options

The options and text for the individual days are stored in macros, the names of which are built up using the days or classnumber for which the option is intended. We will define it using the macro `\options` so that multiple `\options` statements may be used for the same day.

`\ca@addmacro` The first argument is the name of a macro (without the backslash) and the second is a sequence of tokens to be added to its definition. This is taken from the TeXbook, exercise 20.15. Note that the spaces in the last line are essential.

```
\long\def\ca@addmacro#1#2{
 \expandafter\ifx\csname#1\endcsname\relax%
        \expandafter\def\csname#1\endcsname{#2}
  \else
  \toks0=
     \expandafter\expandafter\expandafter{\csname#1\endcsname}
  \toks2={#2}
  \expandafter
    \edef\csname#1\endcsname{\the\toks0 \the\toks2 }\fi}
```

`\options` `\options#1#2` adds the tokens in the second argument to the macro with the name `\csname #1options\endcsname`.

```
\newcommand\options[1]{\ca@addmacro{#1options}}
```

Now the code for the varions options.

`\ifclassday` A switch determines which days are class days.

`\classday`

`\noclassday`

```
\newif\ifclassday
\newcommand{\classday}{\classdaytrue}
\newcommand{\noclassday}{\classdayfalse}
```

`\ifusingmonth` This switch is used in `\ca@doaday` to decide whether the name of the month

`\usingmonthtrue` will be printed every day.

```
\newif\ifusingmonth
```

`\weeklytext` This token box holds the standard text for each day of the week.

```
\newtoks\weeklytext
```

\caltext This macro works like \options to save the text for a specific day, saving the text in a macro with the name \csname #1text\endcsname. We add \par after the text, so successive texts for the same day start on separate lines. We use \par instead of \\ since it is harmless if it is unneeded (but requires that \ca@addmacro be long).

```
\newcommand\caltext[2]{\ca@addmacro{#1text}{#2\par}}
```

textdaycount \caltexton \castextnext The commands \caltexton and \caltextnext use \caltext with the day determined by the counter textdaycount.

```
\newcounter{textdaycount}\setcounter{textdaycount}1
\newcommand\caltexton[2]{\setcounter{textdaycount}{#1}
        \caltext{C#1}{#2}}
\newcommand\caltextnext[1]{\advance\c@textdaycount by 1
        \caltext{C\thetextdaycount}{#1}}
```

## 2.2 Macros concerned with date calculations

Now we have a selection of macros which are concerned with calculating the dates in the calendar.

date Counter for the day of the month.

```
\newcounter{date}
```

month Counter for month (January = 1)

```
\newcounter{month}
```

year Counter for year.

```
\newcounter{year}
```

\curdate Print out the date.

```
\newcommand\curdate{\arabic{month}/\arabic{date}/\arabic{year}}
```

\monthname Print the name of the Month.

```
\newcommand\monthname{\ifcase\c@month\or Jan\or Feb\or Mar\or Apr%
        \or May\or June\or July\or Aug\or Sep\or Oct%
        \or Nov\or Dec\fi}
```

\advancedate Move the date forward by one day.

```
\newcommand\advancedate{\stepcounter{date}
    \ifnum\thedate>\monthlength\relax
       \addtocounter{date}{-\monthlength}\advancemonth\fi}
```

`\ifnewmonth` True if no days have been printed during the current month.

```
\newif\ifnewmonth\newmonthtrue
```

`\advancemonth`

```
\newcommand\advancemonth{%
   \global\newmonthtrue\stepcounter{month}
   \ifnum\c@month>12
      \stepcounter{year}\setleap\setcounter{month}1\fi}
```

`\ifleap` True if the year is a leap year.

```
   \newif\ifleap
```

`\setleap` Determine whether the year is a leap year. Note that 2000 is a leap year, so this is correct until 2100 by which time the next version should be out.

```
\newcommand\setleap{%
   \@tempcnta=\c@year
   \divide\@tempcnta by 4 \multiply\@tempcnta by 4
   \ifnum\@tempcnta=\c@year\global\leaptrue
   \else\global\leapfalse\fi}
```

`\monthlength` Determine the number of days in the current month.

```
\newcommand\monthlength{%
   \ifcase\c@month\or31\or\ifleap29\else28\fi
     \or31\or30\or31\or30\or31\or31\or30\or31\or30\or31\fi%
   \relax}
```

`\setdate` Take argument in the form `m/d/y` and set the counters `month`, `date` and `year`.

```
\newcommand\setdate[1]{\setdate@#1!}
\def\setdate@#1/#2/#3!{
       \setcounter{month}{#1}
       \setcounter{date}{#2}
       \setcounter{year}{#3}
       \global\newmonthtrue\setleap}
```

`\ordinaldate` Print the day of the month as an ordinal.

```
\newcommand\ordinaldate{\ordinal{\c@date}}
```

`\ordinal` Print the contents of a register as ordinal number.

```
\newcommand\ordinal[1]{%
   \let\last@=\relax\let\last@@=\relax
```

```
    \expandafter\@rd\the#1x}
\newcommand\@rd[1]{\ifx#1x\if\last@@1th\else\@rdend{\last@}\fi\else
    \let\last@@=\last@\def\last@{#1}#1\expandafter\@rd\fi}
\newcommand\@rdend[1]{\ifcase#1 th\or st\or nd\or rd\else th\fi}
```