

# Chapter Mottos and Optional Semi-Parameters in General and for L<sup>A</sup>T<sub>E</sub>X

Reinhard Wonneberger  
Hamburg\*

## Abstract

Motto texts will cause some logical and practical difficulties, when they are to be prefixed to chapter headings. To solve them, the motto text should be specified after the chapter heading. To allow for a variety of contained constructs such as footnotes or verbatims, this text must not be read as an ordinary parameter. These antagonistic goals are reconciled in a construct which looks like a parameter but is treated as input text. This concept is a modified version of the technique used for PLAIN footnotes. We give all macros necessary to implement this concept as an extension to L<sup>A</sup>T<sub>E</sub>X.

## 1 On Mottos

To provide the reader with a glimpse of what is waiting for him, a book or its chapters are sometimes prefixed with mottos. The basic idea of mottos, going right into the heart of a text in one short sentence, can be traced back to the times of ancient Babylonia, the myth of *Atramḥasīs* starting with such a motto verse:<sup>1</sup>

*inūma ilū awilum*  
When the gods were (also still) men ...

---

\* The macros presented here were developed at DESY, Notkestraße 85, D 2000 Hamburg 54, FRG. Comments should be sent to R. W., Drachenstieg 5, D 2000 Hamburg 63 or through Bitnet/Earn to B03WBG at DHHDESY3.

<sup>1</sup> Wolfram von Soden: *Mottoverse zu Beginn babylonischer und antiker Epen, Mottosätze in der Bibel*. In: W. v. S.: *Bibel und Alter Orient. Altorientalische Beiträge zum Alten Testament*. Hans-Peter Müller (ed.). Berlin / New York: de Gruyter 1985, p. 206 from p. 206-212.

From a linguistic point of view, mottos are somewhat similar in function to particles, being both part of the text and a comment on it. So they are better understood in terms of a metatext.<sup>2</sup>

There is a wide range of possible motto texts, reaching from witty to ænigmatic, from aphoristic to devotional, from past to present. Normally motto texts will be quotations from some celebrity, but nowadays *graffitti* representing the *vox populi* will also be found.

As far as typesetting is concerned, graphic arrangement of mottos should meet several requirements. The special kind of text will be made clear through emphasis or even a different family of character type, e.g. *sans serif*. A scope, i.e. the range of text the motto applies to, will be expressed by prefixing the motto to an already established unit like a chapter. And finally, æsthetic concepts should be taken into account. So the motto will normally be broken into smaller lines, which may be right-adjusted to stress the frame of the page in connection with a heading. Professional book designers should be consulted on a specific concept for formatting.

Since mottos are normally taken from some source, one might also wish to put names into the index or give some bibliographic information in a footnote. Though there seems to be nothing peculiar about these requirements, implementation of mottos needs

---

<sup>2</sup> It is fascinating to watch the gradual emergence of such metatexts, which are also used to give direct access to texts, a history still waiting to be written. For some remarks, cf. R. W.: *Normaltext und Normalsynopse. Neue Wege bei der Darstellung alttestamentlicher Texte*. Zeitschrift für Sprachwissenschaft 3 (1984) 203-233; cf. also R. W.: *Leitfaden zur Biblia Hebraica Stuttgartensia*. Göttingen: Vandenhoeck & Ruprecht 1984, chapters 3-5.

finding a solution that is upward-compatible. Our macros should also work for an alternative (asterisk) chapter command. A mottochapter macro looking quite similar to a chapter macro might look like this:

```
\mottochapter[toc_entry]{heading}%
{Here comes the motto text,
preferably from Shakespeare
\index{Shakespeare}
or the Bible,\footnote
{Use RSV \index{RSV}
to produce special effects!}
enclosed in braces.}
\par First paragraph of the chapter.
```

The only difference from the normal `\chapter` macro is the additional third parameter.

Since characters are assigned category codes when they are read from the file, and since category codes once set cannot be changed afterwards, our requirements cannot be met when the motto text has to be read as a parameter. What we would need then is the facility to specify the motto text as a parameter, but to read it as ordinary text. This sounds contradictory, but it is nevertheless possible in  $\text{\TeX}$ , and to have a name for this construct, we shall call it a *semi-parameter*.

### 3 Semi-Parameter Footnotes

The concept of semi-parameters has already been realized in the footnote macros described in *The  $\text{\TeX}$ book* p. 363, whereas the restrictions we mentioned above also apply to ordinary  $\text{\LaTeX}$  footnotes, which read the footnote text as a parameter. To avoid these restrictions, we modified the  $\text{\LaTeX}$  footnote macros in this respect. Since it requires some care to adapt the general technique to  $\text{\LaTeX}$  control sequences, it might be helpful to give the actual code here.<sup>8</sup> An explanation of the basic technique will be given in the next section.

8

```
%\head footnotes
% The following macros avoid to read the footnote
% text as parameter. They are taken from PLAIN.
% LaTeX control sequences have been substituted
% for those from PLAIN, marked by %LATEX.

% \footnotesep : The height of a strut placed
% at the beginning of
\footnotesep=0pt
% do not sep footnotes for esthetic reasons
```

Though the technique is basically the same as the one explained before, the semi-parameter here

```
% LaTeX keeps PLAIN TeX's \footnoterule as the default
%\def\footnoterule{\kern-3\p@
% \hrule width 2truein \kern 2.8\p@}
% raise fnrule for esthetic reasons
\def\footnoterule{\kern-3.9\p@
\hrule width 2truein \kern 3.5\p@}

% \newinsert\footins
%\def\footnote#1{\let\@sf\empty
% \ifhmode\edef\@sf{\spacefactor\the\spacefactor}}\fi
% #1\@sf\vfootnote{#1}}
% \def\vfootnote#1{\insert\footins\bgroup
\def\@footnotetext{\insert\footins\bgroup
\footnotesize %LATEX
\interlinepenalty\interfootnotelinepenalty
% \splittopskip\ht\strutbox % top baseline for broken f.
\splittopskip\footnotesep %LATEX
\splittmaxdepth\dp\strutbox \floatingpenalty\@MM
% \leftskip\z@skip \rightskip\z@skip \spaceskip\z@skip
% \xspaceskip\z@skip
\hsize\columnwidth \parboxrestore %LATEX
\edef\@currentlabel{\csname p@footnote\endcsname %LATEX
\@thefnmark}% %LATEX
% \textindent{#1} %Knuth
% \textindent{\@thefnmark} % yields normal numbers
\@makefnlabel
\footstrut\futurelet\next\fo@t}

% the following is an addition to REPORT.sty
% format the footnote label inside the footnote
\def\@makefnlabel{\@par \parindent 1em\noindent
\hbox{\lower0.25ex\hbox{\quad$\sim\@thefnmark$\quad}}}

% \long\def\@footnotetext#1{\insert\footins{\footnotesize
% \interlinepenalty\interfootnotelinepenalty
% \splittmaxdepth\dp\strutbox \floatingpenalty \@MM
% \hsize\columnwidth \parboxrestore
% \edef\@currentlabel{\csname p@footnote\endcsname
% \@thefnmark}
% \@makefnlabel
% {\rule{\z@}{\footnotesep}}\ignorespaces
% #1\strut}}

\def\fo@t{\ifcat\bgroup\noexpand\next \let\next\fo@t
\else\let\next\fo@t\fi \next}
\def\fo@t{\bgroup
\rule{\z@}{\footnotesep}\ignorespaces %LATEX
\aftergroup\@foot\let\next}
\def\fo@t#1{#1\fo@t}
% \def\@foot{\strut\egroup}
\def\@foot{\strut\egroup
\ifhmode\spacefactor\@x@sf\relax\fi} %LATEX

\def\footstrut{\vbox to\splittopskip{}}
\skip\footins=\bigskipamount % space added
% when footnote is present
\count\footins=1000 % footnote magnification factor
\dimen\footins=8in % maximum footnotes per page

\def\fo@t{\footnote\bgroup} \def\ef@t{\egroup} % local defs
```

is *obligatory*, and the next token will be used if no proper parameter is found.

The use of semi-parameters is a first step towards the solution of our motto problem. If we can also make this parameter optional, we can even achieve compatibility with earlier texts and save the effort to type empty motto parameters if no motto is present. This, too, is possible in T<sub>E</sub>X, and so we arrive at the concept of *optional semi-parameters*.

It should be noted, however, that this concept is not in accordance with the rules of L<sup>A</sup>T<sub>E</sub>X, which require optional parameters to be enclosed in brackets ([. . .]). But this means parameter reading, just the thing we want to avoid. On the other hand, optional and also obligatory semi-parameters do not interfere with L<sup>A</sup>T<sub>E</sub>X syntax, and in some cases are even an enhancement.

## 4 Optional Semi-Parameters

As we said before, the technique of semi-parameters is shown in the footnote handling macros of *The T<sub>E</sub>Xbook* p. 363, but it is not explained there. Because this technique is of general importance, we should drag it out of the discouraging bunch of footnote submacros, and present it here with a few explanations. Our version will be different from the original one in two respects. First, to avoid conflicts with possible footnotes, we use the same code with different names; second, we modify the macros so that they will test whether the semi-parameter is present or not, thus allowing it to be optional instead of required as with footnotes.

```
\def\@readmotto{\begingroup \mottoformat
\futurelet\next\mo@t} % 1
```

```
\def\mo@t{\ifcat\bgroup\noexpand\next
\let\next\m@@t
\else\let\next\m@t\fi
\next} % 2
```

```
\def\m@@t{\bgroup\aftergroup\@motto
\let\next} % 3
```

```
\def\m@t{\nomottoformat \endgroup
\@aftermotto} % 4
```

```
\def\@motto{\endmottoformat \endgroup
\@aftermotto} % 5
```

```
\def\@aftermotto{\%
\@printmottoheading{\@headingtext}
\@afterheading
\if@twocolumn \@endtopnewpage \fi} % 6
```

1. The first macro, after starting a group and the formatting environment for mottos, does nothing but load the next element into `\next` for inspection by the second macro. This element is not removed from the input. To make sure the outside world is not affected by our operations, we enclose everything to follow in a 'hard' group using the `\begingroup` primitive, which must be closed by an `\endgroup` primitive and so might help to detect grouping errors in the motto text.
2. The second macro will decide between a group (*if* case) and any other context (*else* case), choosing the latter also if the group follows after a space. This allows us to follow a mottoless `\chapter` macro with a normal group.
3. The third macro (*if* case) first opens a group to replace the group opening symbol that has been found by `\futurelet` before and will be swallowed at the end of this macro by `\let` (see *The T<sub>E</sub>Xbook* p. 376). This is necessary because the `\aftergroup` action must be defined from *within* the group after which it is to be performed. Then it stores the macro to be executed after the group. This group will be the motto text semi-parameter read from input, and accordingly it will be closed by the group closing symbol of the motto text coming from the input. Finally it swallows the group opening symbol from the input text, in order to compensate for the group opening symbol it put in before.
4. The fourth macro (*else* case) is executed when no group follows immediately. It contains a macro that will typeset something other than the missing motto, typically some sinkage (vertical space). It closes the 'hard' group and executes the `\aftermotto` macro.
5. The fifth macro supplies the corresponding end command for formatting, plus a group clos-

ing symbol for the ‘hard’ group opened in the first macro, and then executes the `aftermotto` actions.

6. The last macro contains the actions to be taken after the motto. If we are in `twocolumn` style, we should close here the corresponding page-wide heading box.

## 5 Formatting the Motto

These macros will execute a formatting macro at the beginning and at the end of the motto. *Leslie Lamport*, who read a draft of this article, strongly recommends that a professional book designer be consulted on the actual formatting of the motto. So, the following formatting is only meant as a default for draft purposes, and the corresponding macros have been given accessible names so that they can be easily redefined by the user.

```
% motto formatting example
\def\mottoformat{\hfill
%           position motto to the right
  \begingroup
  \minipage{0.63\textwidth}
%           narrow motto column
  \parindent 0pt
%           do not indent paragraphs
  \begingroup
  \flushright      % rightadjust text
  \mottofont}
%           use special font, e.g. \sf

\long\def\endmottoformat{%
  \endflushright
%           end rightadjustment of text
  \endgroup
%           \nobreak % penalty against page break
  \endminipage
  \endgroup
  \par}

\def\nomottoformat{%
  \vspace*{10ex} % produce sinkage here
  \relax }

\let\mottofont= \em
%           default for motto font
```

The formatting of the motto text is defined here using only constructs of standard L<sup>A</sup>T<sub>E</sub>X. It can also be used as a separate environment to format a motto without connection to a chapter, which can be useful to test the best shaping of motto texts. To allow this, the normal form of L<sup>A</sup>T<sub>E</sub>X environments, `\begin{flushright} ... \end{flushright}`, should be replaced by the internal form enclosed in a hard group to preserve the original structuring, e.g. `\begingroup \flushright ... \endflushright \endgroup`.

When footnotes are used inside the motto, they will be placed into the box made by the `minipage` environment. This `minipage`-default was chosen because it will work also in the case of `twocolumn` style, where footnotes would otherwise just disappear. Even then, the technique of splitting a footnote into a ‘callout’ and the actual note can be used to obtain ordinary bottom footnotes. In many cases it will be best to specify linebreaks explicitly inside the motto with `\\` (newline) commands. Then it will be appropriate to leave out the `minipage` environment. Examples for both cases will be given in the appendix.

There is also a `\nomottoformat` macro, which allows us to specify appropriate actions if no motto is present. The default will be to add some vertical space (*sinkage*) instead of the motto.

## 6 Upgrading the L<sup>A</sup>T<sub>E</sub>X chapter command

If the chapter macros of L<sup>A</sup>T<sub>E</sub>X are studied carefully, it turns out that motto printing has to be done after the test for `twocolumn` style has been performed in the `\@chapter` and `\@schapter` macros, and before the group for printing the heading is started in the `\@makechapterhead` and `\@makeschapterhead` macros. In fact, the motto production should replace the command immediately before this group, which is a `\vspace*{...pt}` control sequence. This command produces the so-called *sinkage*. There is also a close graphic connection between motto and sinkage: if a motto is present and printed on the right part of the page, a natural sinkage is produced, and the vertical distance between chapter heading and motto can be much smaller than otherwise.

However, the problem cannot be solved just by changing this particular spot. The remainder of

the macro has to be removed and associated with the macro that will end a motto. Things are even more complicated for twocolumn style, which uses a `\@topnewpage[...]` macro that introduces another level of nesting. Though our postponing technique might be used again, it seems easier to split the `\@topnewpage[...]` in a beginning and end part, and integrate it into the `\@makechapterhead` and `\@makeschapterhead` macros.<sup>9</sup>

We also have to make sure that our inspecting macro `\@readmotto` will be the last thing to be executed in the macro it is called from. Thus we have to modify the `\if ... \fi` construction at the end of the `\chapter` macro to avoid calling it from within such a construction.<sup>10</sup> Another important technique

---

9

```

% for normal section %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\def\@makechapterhead#1{\def\@headingtext{#1}\let
  \@printmottoheading=\@makemottochapterhead
  % \tracingall
  \if@twocolumn \@begintopnewpage \fi
  \@readmotto}

\def\@makemottochapterhead#1{\vspace*{3ex}
  { \parindent Opt \raggedright
    \ifnum \c@secnumdepth > \m@ne
  %
    \huge\bf
    \@chapapp{ } \thechapter
  %
    \par
    \vskip 20pt \fi
  %
    \Huge \bf
    #1\par
    \nobreak % TeX penalty to prevent page break.
    \vskip 40pt } } % Space between title and text.

% for star-section %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\def\@makeschapterhead#1{\def\@headingtext{#1}\let
  \@printmottoheading=\@makesmottochapterhead
  \if@twocolumn \@begintopnewpage \fi
  \@readmotto}

\def\@makesmottochapterhead#1{\vspace*{3ex}
  { \parindent Opt \raggedright
    \Huge \bf
    #1\par
    \nobreak % TeX penalty to prevent page break.
    \vskip 40pt } } % Space between title and text.

```

10

```

\def\chapter{\clearpage % Starts new page.

```

that could be used is to assign the next action to be taken to an intermediate control sequence (`\next`) as explained in *The T<sub>E</sub>Xbook* p. 352.

To use the new technique, we can include the new macros and `\let \chapter= \mottochapter`. Previous texts can be run without disturbance, and even when run without our macros, the new input will do no harm: the motto text will just print as normal text after the chapter heading.

## 7 Conclusion

Our discussion might have a whiff of “Much Ado about Nothing”, were it not for the general impor-

---

```

\thispagestyle{plain}
%
% Page style of chapter page is 'plain'
\global\@topnum\z@ % No figures at top of page.
\@afterindentfalse
%
% No indent in first paragraph, otherwise
% change to \@afterindenttrue
\secdef\@chapter\@schapter}

\def\@chapter[#1]#2{\ifnum \c@secnumdepth > \m@ne
  \refstepcounter{chapter}
  \typeout{\@chapapp\space\thechapter.}
  \addcontentsline{toc}{chapter}{\protect
    \numberline{\thechapter}#1}\else
  \addcontentsline{toc}{chapter}{#1}\fi
  \chaptermark{#1}
% Add between-chapter space to lists of figs & tables:
\addtocontents{lof}{\protect\addvspace{10pt}}
\addtocontents{lot}{\protect\addvspace{10pt}}

%begin of modifications: %%%%%%%%%%%%%%%%%%%%%%%%%%%
\@makechapterhead{#2}}
% will replace:
%
% \if@twocolumn % Tests for two-column mode.
%
% \@topnewpage[\@makechapterhead{#2}]
%
% \else \@makechapterhead{#2}
%
% \@afterheading % Routine called after
% \fi} % chapter and section heading.

\def\@schapter#1{\@makeschapterhead{#1}}

% split \long\def\@topnewpage[#1]{...} in two parts: %%%
\def\@begintopnewpage{\@next\@currbox\@freelist{}}{
  \global\setbox\@currbox\ vbox\bgroup
  \hsize\textwidth \@parboxrestore}

\long\def\@endtopnewpage{\par
  \vskip -\dbltextfloatsep\egroup
  \global\count\@currbox\tw@
  \global\@dbltopnum\@ne
  \global\@dbltoproom\maxdimen\@addtodblcol
  \global\vsizel\colht \global\@colroom\@colht}

```

tance of the techniques that it leads to. First of all, there is the technique of *semi-parameters*, which we just borrowed from *The T<sub>E</sub>Xbook*.

Semi-parameters will allow us to perform some actions *after* they have been processed, resembling normal parameters in this respect. But as they are not read like normal parameters, they do not impose restrictions on the type of constructions that they may contain. Thus they allow nesting of “dangerous” operations; for example, the index macro for our book *Verheißung und Versprechen*, which requires special treatment of category codes, can be part of a footnote which is in turn part of a motto. On the other hand, obligatory semi-parameters do not interfere with L<sup>A</sup>T<sub>E</sub>X syntax. As is shown by the `\footnote` case, they are an enhancement and upward-compatible at the same time.

Second, we found a way to make this type of parameter *optional*. As we noted before, this concept is not in accordance with the rules of L<sup>A</sup>T<sub>E</sub>X, which have either obligatory normal parameters or optional parameters enclosed in brackets ([...]). But in both cases parameter reading is implied, just the thing we want to avoid. Though not much harm is done, texts with optional semi-parameters are no longer compatible with standard L<sup>A</sup>T<sub>E</sub>X, unless the corresponding macros are also included. But since they might prove useful in other cases too, and existing L<sup>A</sup>T<sub>E</sub>X input is not affected, it might be worthwhile to consider them for the next major release. And I should like to add that in my opinion L<sup>A</sup>T<sub>E</sub>X is too important a tool to be frozen already in its present state.

In coming to the end of this article, we should give some thought to its history. When it became clear to me that the motto problem was not trivial, I did not start by writing macros to solve the problem, but by writing explanations for macros that did not exist at that time. The macros were written only after things had become clear to me through the process of compiling the explanations. I am quite convinced that this saved me more testing time than it cost me in writing, while giving other people access to the concepts of these macros at no additional cost and thus saving them the burden of exploring ways that lead to nowhere.

This approach was supported by the file inclusion mechanism of our operating environment, which allows us to include a macro everywhere. So we can use the same source code to be executed and

to be printed as a verbatim listing. As far as I can see, this is not possible in L<sup>A</sup>T<sub>E</sub>X, but might be implemented using the technique described in *The T<sub>E</sub>Xbook* p. 380f.

Modifying existing macros and writing new ones could be improved a great deal if, instead of some sparse comments in the source files, we had a WEB-like style of macro development and description, and this article should be seen as a first attempt to move towards a technique that has proven to be one of the most successful tools in software engineering.

*On aurait souhaité de n'être pas technique.  
A l'essai, il est apparu que,  
si l'on voulait épargner au lecteur les détails précis,  
il ne restait que des généralités vagues,  
et que toute démonstration manquait.*

ANTOINE MEILLET,  
Esquisse d'une histoire  
de la langue latine, 1928.<sup>11</sup>

## Appendix A

# Testing Mottos

Our macros are based on the `\chapter` command, which is not present in the article style used here. In order to test them, we first have to include the macros for chapters from the `rep12.sty` or a similar file and then our own, so that a complete set of macros will emerge. Then we `\let \mottochapter= \chapter` and give our text, which is shown in the footnote to our example motto.

Note that we used motto formatting without a minipage environment, controlling the linebreaks explicitly according to the parts of the sentence.<sup>12</sup>

<sup>11</sup> Quoted from CURTIUS LITERATUR 7. — Note that even verbatim text can be contained in a footnote being part of a motto:

```
\chapter{Testing Mottos}%
{On aurait souhaité de n'être pas technique.\\
A l'essai, il est apparu que, \\
si l'on voulait épargner au lecteur les détails précis,\\
il ne restait que des généralités vagues,\\
et que toute démonstration manquait.\\
\em {\sc Antoine Meillet,}\\
Esquisse d'une histoire\\ de la langue latine, 1928.\footnote
{Quoted from {\sc Curtius Literatur} 7.%
}}}% end of footnote, end of \em, end of motto
```

<sup>12</sup> This principle was used to typeset a whole book on books for children; an example is reproduced in my article *Normaltext und Normalsynopse* (see above) in Fig. 20 on p. 225.

To close our article, we use a `mottoformat` environment containing a `minipage` environment<sup>13</sup> to highlight the words found at the end of the account of *Sinuhe*:

*It has come (to its end)  
from beginning to end  
as it had been found in writing.<sup>a</sup>*

---

<sup>a</sup>James B. Pritchard: *Ancient Near Eastern Texts Relating to the Old Testament*. Princeton 1969, p. 23.

---

13  
`\begin{quote}`  
`\begin{mottoformat}`  
 It has come (to its end)\  
 from beginning to end\  
 as it had been found in writing.\footnote  
 {James B. Pritchard:  
 Ancient Near Eastern Texts  
 Relating to the Old Testament.  
 Princeton 1969, p. 23.}  
`\end{mottoformat}`  
`\end{quote}`