The `textcmds` package

Michael J. Downes
American Mathematical Society

Version 1.06, 2007/10/19

1 Introduction

The `textcmds` package provides shorthand commands for all the text symbols that are traditionally produced in LaTeX documents by non-letter ligatures. One of the principal benefits of using these commands is that it makes translating your document from LaTeX to some other form (e.g., HTML) easier and less bug-prone. But it also makes your document less dependent on the use of special font metric files having the required ligature information, and it makes it far easier to achieve special effects for the characters in question—for example, to add or not to add a small amount of extra space around an em-dash character. With the ligature method you have to manually add the space for each instance, whereas if you use the `\mdash` command, it suffices to change the definition of `\mdash` to suit your wishes.

All of these definitions use the preferred font-encoding-independent LaTeX commands to obtain the characters in question.

Command	**Definition**	**Result**
`\mdash`	`\textemdash`	—
`\ndash`	`\textendash`	–
`\qd`	`\textquestiondown`	¿
`\xd`	`\textexclamdown`	¡
`\ldq`	`\textquotedblleft`	“
`\rdq`	`\textquotedblright`	”
`\lq`	`\textquoteleft`	‘
`\rq`	`\textquoteright`	’

This package also provides short forms for certain text symbols whose generic name is too long for convenient entry. (The `\cwm` command does not produce visible output but marks word boundaries in a compound word.)

Command	Definition	Result
\bul	\textbullet	•
\vsp	\textvisiblespace	␣
\pdc	\textperiodcentered	·
\vrt	\textbar	|
\cir	\textasciicircum	^
\til	\textasciitilde	~
\bsl	\textbackslash	\
\cwm	\textcompwordmark	

Finally, a few other miscellaneous commands are provided, including a quoting command \qq. It seems clearly consonant with other parts of LaTeX to write \qq{...} to quote a word or short phrase rather than \ldq ...\rdq; and the use of higher-level markup is groundwork that must be laid if one should ever want to do anything more sophisticated at the boundaries of a quoted expression (such as automatically transposing the quote character with following punctuation, if traditional rather than logical punctuation style is desired).

Example	Definition	Result
\qq{some text}	\ldq#1\/\rdq	“some text”
\q{some text}	\lq#1\/\rq	‘some text’
\lara{some text}	\textlangle#1\textrangle	⟨some text⟩
Jello\tsup{TM}	*raise .9ex*{\supsize#1}	Jello$^{\mathrm{TM}}$
Jello\tsub{TM}	*lower .6ex*{\supsize#1}	Jello$_{\mathrm{TM}}$
a\tprime b	\tsup{\textprimechar}	a′b

⟨∗pkg⟩

2 Implementation

Package name, date, version number.

```
\ProvidesPackage{textcmds}[2007/10/19 v1.06]
```

Dashes and inverted beginning-of-sentence punctuation.

```
\providecommand{\mdash}{\textemdash\penalty\exhyphenpenalty}
\providecommand{\ndash}{\textendash\penalty\exhyphenpenalty}
\newcommand{\qd}{\textquestiondown}
\newcommand{\xd}{\textexclamdown}
```

Quote commands. Note that \lq and \rq are defined in the LaTeX kernel to produce functionally different quote characters.

```
\newcommand{\ldq}{\textquotedblleft}
\newcommand{\rdq}{\textquotedblright}
\newcommand{\lsq}{\textquoteleft}
\newcommand{\rsq}{\textquoteright}
```

```
\newcommand{\bul}{\textbullet}%
\newcommand{\vsp}{\textvisiblespace}%
\newcommand{\pdc}{\textperiodcentered}%
\newcommand{\vrt}{\textbar}%
\newcommand{\cir}{\textasciicircum}%
```

```
\newcommand{\til}{\textasciitilde}%
\newcommand{\bsl}{\textbackslash}%
\newcommand{\cwm}{\textcompwordmark}%

\providecommand{\qq}[1]{\ldq#1\/\rdq}

\providecommand{\q}[1]{\lq#1\/\rq}
```

Unlike `\textsuperscript` and `\textsubscript`, these do not use math mode at all. The difference between `\scriptsize` and `\supsize` is that the former is fixed at a single constant size regardless of context, whereas the latter adapts to the current font size.

```
\newcommand{\supsize}{%
```

Cf `\glb@settings`.

```
    \expandafter\ifx\csname S@\f@size\endcsname\relax
      \calculate@math@sizes
    \fi
    \csname S@\f@size\endcsname
    \fontsize\sf@size\z@\selectfont
  }
  \DeclareRobustCommand{\tsup}[1]{%
    \leavevmode\raise.9ex\hbox{\supsize #1}%
  }
  \DeclareRobustCommand{\tsub}[1]{%
    \leavevmode\lower.6ex\hbox{\supsize #1}%
  }
```

The LaTeX kernel contains fallback definitions for various symbols that traditionally came from the `cmsy` font:

```
\DeclareTextSymbolDefault{\textbraceleft}{OMS}
```

But there is no definition of that kind for the cmsy prime character that we want to use for `\tprime`. So we need to do it here.

```
\DeclareTextSymbolDefault{\textprimechar}{OMS}
\DeclareTextSymbol{\textprimechar}{OMS}{48}
\DeclareRobustCommand{\tprime}{\tsup{\textprimechar}}
```

And one more pair of symbols that are sometimes useful in text, yet do not have suitable text definitions in the LaTeX kernel. (They do in the `textcomp` package.)

If the `textcmds` package is loaded together with the `textcomp` package, we don't want to clobber the TS1 default.

```
\@ifundefined{textlangle}{%
  \DeclareTextSymbolDefault{\textlangle}{OMS}
  \DeclareTextSymbolDefault{\textrangle}{OMS}
}{}
\DeclareTextSymbol{\textlangle}{OMS}{"68}
\DeclareTextSymbol{\textrangle}{OMS}{"69}
```

Not sure what's the best name for the angle-brackets analog of the `\qq` command. How about "lara" for "left-angle right-angle"?

Unlike the quotes case, it is highly unlikely that the font contains kern information for the rangle character and the character preceding it! So let's put in an italic correction.

```
\DeclareRobustCommand{\lara}[1]{\textlangle#1\/\textrangle}

\csname endinput\endcsname
```
⟨/pkg⟩

Do you want some Emacs code to convert `--` to `\ndash` while you write? And ‘‘ to `\qq{` ? Try this.

```
<*emacs>
(defvar latex-ndash-command "\\ndash"
  "*String to insert for an n-dash in LaTeX mode.")

(defvar latex-mdash-command "\\mdash"
  "*String to insert for an m-dash in LaTeX mode.")

(defvar latex-quote-command "\\qq"
  "*String to insert for quotes in LaTeX mode.")

(defun latex-maybe-start-quotes (arg)
  "Insert the beginning of a \\qq{...} structure if the preceding char is
a left quote."
  (interactive "*p")
  (if (= (preceding-char) ?\‘)
      (progn
        (delete-backward-char 1)
        (insert-and-inherit (concat latex-quote-command "\{")))
    (self-insert-command arg)))

(defun latex-maybe-end-quotes (arg)
  "Insert the end of a \\qq{...} structure if appropriate."
  (interactive "*p")
  (if (= (preceding-char) ?\’)
      (progn
        (delete-backward-char 1)
        (insert-and-inherit "\}"))
    (self-insert-command arg)))

(defun latex-maybe-dash (arg)
  "Convert two or three hyphens to \\mdash or \\ndash."
  (interactive "*p")
  (cond
   ((re-search-backward
     (concat (regexp-quote latex-ndash-command) " *\\=") nil t)
    (replace-match (concat (regexp-quote latex-mdash-command) " ")))
   ((= (preceding-char) ?-)
    (delete-backward-char 1)
    (insert-and-inherit (concat latex-ndash-command " ")))
   (t (self-insert-command arg))))
```

```
(add-hook 'TeX-mode-hook
  '(lambda
    (define-key LaTeX-mode-map "\`" 'latex-maybe-start-quotes)
    (define-key LaTeX-mode-map "\'" 'latex-maybe-end-quotes)
    (define-key LaTeX-mode-map "-" 'latex-maybe-dash)))
</emacs>
```