The titlecaps Package

Routines for setting rich-text input into Titling Caps

Steven B. Segletes
SSegletes@verizon.net

2022/04/12
v1.3

1 Description and Commands

The titlecaps package is intended to be used to convert lower-cased text into titling caps, wherein the first letter of every word is capitalized, except for words designated to remain in lower case, for example prepositions and conjunctions. While this function has been performed elsewhere, for example in the stringstrings package, it is here significantly enhanced. The stringstrings implementation of titling caps is particularly limited because of its slow speed (I criticize stringstrings because I wrote it). Furthermore, that implementation only works on textual expressions, which do not contain any font-modification commands.

In contrast, the titlecaps package is set up to work in conjunction with font-modification commands that change the family, series, or shape of the font. Likewise, it will work with the fontsize changing commands (`\tiny`, `\scriptsize`... `\Huge`) embedded in the argument. It is also, unlike the stringstrings version, able to screen out most punctuation when deciding whether a word is predesignated as lower case. Furthermore, it looks for symbols that might signify the beginning of a new text "group", such as (, [, {, ‘, and -, and thereafter titles the word that follows, even though the first alphabetic letter is not technically at the lead of the "word."

The primary commands that have been implemented in this package are the following:

`\titlecap[`*option*`]{`*rich text*`}`
`\Addlcwords{`*designated lower-case space-separated word list*`}`
`\Resetlcwords`

In addition, the following auxiliary commands are also available:

`\textnc{`*text*`}`
`\def\converttilde{`T *or* F`}`
`\noatinsidetc`
`\usestringstringsnames`

\titlecap The \titlecap command is the primary contribution of this package. It will (within constraints) capitalize the first letter of each word in the argument. The primary constraint under which it operates is the existence of a predefined
\Addlcwords lower-cased word list that is set by the user, using the \Addlcwords command. These predefined lower-cased words are passed as an argument to \Addlcwords in a space-separated list. Subsequent invocations add to the existing list of predefined lower-cased words. The user may clear the list of predefined lower-
\Resetlcwords cased words through the issuance of the \Resetlcwords command.

When \titlecap is invoked, it first breaks the argument into individual pieces that constitute "words." Note, however, that these "words" may include punctuation and text formatting commands interspersed with the actual text. This is an essential challenge to overcome.

After breaking the argument into words, \titlecap will attempt to match each word of the argument to the list of predesignated lower-cased words. In order to assist this process, the command (temporarily) screens out text-formatting commands and punctuation marks from the argument, so that their presence does not inhibit a word match with the lower-cased word list (see Quirks, Tricks, and Limitations for exceptions). It flags matches, so that these matching words will not later be titled.

The command will then reconstitute the "words" of the argument with their punctuation and font-changing commands intact. In the default invocation, \titlecap will capitalize the first word of the argument, even if that word is on the lower-cased word list. This default may be overridden with the command's optional argument. Employing anything other than a capital "P" will treat the first word of the argument like any other word, meaning it will be title-capped only if it does not appear on the predesignated lower-cased word list.

The algorithm that searches the "words" of the argument character by character, in order to titlecap the first letter of a word, is, generally speaking, able to process only textual content. However, special provisions have been enacted to handle the following text-formatting commands within the argument of \titlecap:

\textup	\upshape	\tiny	\Huge
\textit	\itshape	\scriptsize	\textnc[a]
\textsc	\scshape	\footnotesize	
\textsl	\slshape	\small	
\textmd	\mdseries	\normalsize	
\textbf	\bfseries	\large	
\textrm	\rmfamily	\Large	
\textsf	\sffamily	\LARGE	
\texttt	\ttfamily	\huge	

[a] This command is introduced by this package.

Other macros can generally be handled in the argument to `\titlecap` only if they expand to textual content.

Once the first word of the argument has been titled "by force" (or not, with the use of the optional argument), the command proceeds through each word, deciding whether or not to title the word, based on the lower-cased word flag that has previously been set. The titlecaps package can notably handle the titling of strings containing both diacritical marks found in various languages (such as ò, ó, ô, ö, õ, ō, ȯ, ǒ, ŏ, ő, o͡o, o̩, o̩, and o̲), as well as national symbols (such as œ, æ, and ø [see Quirks, Tricks, and Limitations]).

While punctuation had been earlier screened out in order to search for predefined lower-cased words, that is a slightly different problem from that of figuring out how to titlecap a punctuated word not on the lower-cased list. While many punctuation marks trail a word and are, therefore, not a problem, several punctuation marks lead a word, or indicate a group separator, even in the absence of whitespace. `\titlecap` must make sure that these sorts of punctuation marks do not inhibit the capitalization of the subsequent letter. To this end, `\titlecap` looks for instances of the following five characters: -, (, [, {, and ‘, and flags the next character for possible capitalization (unless the word had been previously identified as predefined lower-cased [see Quirks, Tricks, and Limitations]).

2 Quirks, Tricks, and Limitations

While titlecaps has been set up to run with certain embedded size and font-changing commands, it will not, in general, work with macros in the argument, unless the macros expand directly to a text string.

The titlecaps package is designed to work with diacritical marks (for example, umlauts) as well as national symbols (symbols like æ, œ, *etc.*). There remain, however, two national symbols which are not handled properly by this package. They are å and ɬ. They will not be capitalized, even if found at the beginning of a word.

The titlecaps package is designed to screen out punctuation when searching for words that are pre-designated as lower-cased. So, for example, the word (*if*) or *“if”* or [*if* or *if,* will all be found to match *if* when it is predesignated as lower cased. However, titlecaps cannot screen out the curly braces `\{` and `\}` from the punctuation list. Thus, {if} will be capitalized as {If} by titlecaps. A workaround is detailed in each of the next two paragraphs.

To prevent a word from being titled (to force it into lower case), it can be immediately preceded by a `\relax`. In this way, the `\relax` is titled, rather than the following word. This method can be used to for one-time exceptions to titling,

or to overcome the curly-brace problem described above, as in `\{\relax if\}`.

`\textnc` The package introduces a command, `\textnc` (standing for "text no-change"). If used outside of the `\titlecap` argument, it is defined as `\def\textnc#1{#1}` and so is leaves the argument intact. Inside of the `\titlecap` argument, however, it forces its argument to be independently considered as text, regardless of any surrounding punctuation or other characters. Thus, this approach may also be used to address the curly-brace issue as `\{\textnc{if}\}`. In this case, "if" would be titled if it is not on the lower-cased word list, but left in lower case if it were on the lower-cased word list. The `\textnc` command is useful in a number of ways inside the argument to `\titlecap`, but always to signify its argument is to be treated as a block of text, independently evaluated for its predesignated lower-cased content.

If a separator like, let's say, a left paren, is used without whitespace, in the fashion of "`a(b)`," then neither `a` nor `b` could possibly be detected as lower-cased words, since the "word" without punctuation would be `ab`, which is neither `a` nor is it `b`. For individual instances or exceptions, the insertion of a `\relax` prior to "`a`" or "`b`" would prevent titling of these terms. Alternately, the word "`ab`" could be added to the predesignated lower-cased words list, in which case, "`a(b)`" would be preserved in lower case. A third (and perhaps preferred) method is the use of the `\textnc{}` construct, which is introduced solely for the purpose of designating embedded text as a separate word grouping. Thus, `\textnc{a}(\textnc{b})` as an argument to `\titlecap` would guarantee that both "`a`" and "`b`" would be independently examined for their presence in the lower-cased word list, irrespective of any surrounding punctuation. In this paragraph, the letters "`a`" and "`b`" have been used for simplicity, but could actually represent words or groups of words.

By LaTeX convention, expansion of the construct "`very \large big`" will associate the `\large` as the first letter of `big`, rather than as the last letter of `very`. Unfortunately, leaving it that way will screw up the titling of `big`. Thus, when adapting the use of fontsize changes to the titlecaps package, the prior space is unskipped, and a space is added after the fontsize change invocation, so that the fontsize change command is at the end of the prior word, rather than at the beginning of the next word. The one adverse side effect to this approach is that a space **will** appear after a fontsize command, even if one is not desired (for example, changing font size just prior to punctuation). One can either issue a `\unskip` following the fontsize change to back-gobble the newly introduced space, or else place the fontsize change following the punctuation.

`{{ }}` While `\uppercase` will not work within the argument of a `\titlecap`, it was found that enclosing part of the argument in double braces will produce upper case for that double-enclosed text. Thus,

```
\titlecap{This is a {{v\"ery}} big test}
```

will produce This is a VËRY Big Test.

The implementation to make the fontsize change commands work within the arguments of `\titlecap` required the use of `\makeatletter` within `\titlecap` itself (which is revoked with a `\makeatother` upon exit from `\titlecap`). If that is not considered a "good practice" for your situation, then you can disable this feature with the command `\noatintc`. However, it is likely, at that point, that the fontsize changing commands as arguments will break the `\titlecap` command.

\noatintc

Except in the case of the various `\text`*xx*`{}` commands, for which special provision has been made, and for the double-brace quirk mentioned above, the argument of `\titlecap` should not contain braces used as group delimiters, for example, in the fashion of:

```
\titlecap{this is a {\itshape test of
  the} emergency broadcast system}
```

It may not break the code, but will likely not produce a desired result. If groupings must be placed in the argument, using `\textnc{}` to condition the argument is recommended. Thus,

```
\titlecap{this is a \textnc{\itshape test of
  the} emergency broadcast system}
```

will result in "This is a *Test of the* Emergency Broadcast System" (of course, in this case, a `\textit{}` would have worked directly, without a problem).

There should be no direct use or need to nest `\titlecap` commands. However, if it is absolutely necessary, the embedded invocation of `\titlecap` should be expressed as

`\titlecap{...\titlecap\textnc{...}...}`.

3 Auxiliary Commands

In addition to the primary commands offered by this package, there are several auxiliary commands.

As described in the section Quirks, Tricks, and Limitations, the `\textnc` command was introduced when additional grouping must occur inside the argument of a `\titlecap` command. The command `\textnc` will itself make no changes to its argument, but will force `\titlecap` to independtly evaluate the `\textnc` argument for its lower-cased content, regardless of any surrounding punctuation or text. The command may only be *needed* within an argument to `\titlecap`. However, if your application requires its invocation outside of a `\titlecap` argument, it will merely output the argument without any modification.

\textnc

By default, the titlecaps package treats hardspaces (~) as characters and not white space. This default treatment can be changed by setting the following parameter: `\def\converttilde{T}`. Following that invocation, hardspaces in

\converttilde

the argument of \titlecap should be indistinguishable from white space.

\noatinsidetc The command \noatinsidetc was briefly described in Quirks, Tricks, and Limitations. Its invocation will prevent the \titlecap command from employing the \makeatletter command, if this is considered bad practice for your setting. However, its invocation will likely make the titlecaps package unable to process fontsize command changes.

\usestringstringsnames The last of these auxiliary commands is \usestringstringsnames. This command will redefine certain command names previously defined by the stringstrings package to instead use the corresponding commands of the titlecaps package, in essence redirecting the stringstrings invocation to the current package. The redirected commands include the following (note: see the stringstrings package documentation for command arguments and options):

```
\addlcwords
\resetlcwords
\addlcword
\getargs
\capitalizetitle
```

If this command is invoked without the prior loading of the stringstrings package, these commands will still be enabled for subsequent use. Because the titlecaps package produces output which can include font changes and other material that cannot be placed into an \edef, the output of the quiet version of the newly redirected \capitalizetitle[q] produces a \def\thestring{} rather than an \edef\thestring{}.

4 Acknowledgements

I would like to acknowledge the assistance of David Carlisle who, through the tex.stackexchange website, assisted with both the string parsing and punctuation screening techniques of this package:

http://tex.stackexchange.com/questions/101604/parsing-strings-containing-diacritical-marks-macros

http://tex.stackexchange.com/questions/105735/ignoring-punctuation-during-comparison

I would also like to thank Prof. Enrico Gregorio for his discovery of several bugs in the package and suggestions for fixes:

http://tex.stackexchange.com/questions/225434/how-to-workaround-conflict-between-greek-and-titlecaps

5 A \titlecap Demonstration for Beginners, Expressed in \titlecap

To Know That None of the Words Typed in This Paragraph Were Initially Upper Cased Might Be of Interest to You. It is Done to Demonstrate the Behavioral Features of This Package. First, You Should Know the Words That I Have Pre-Designated as Lower Case. They Are: "for a is but and with of in as the etc on to if." You Can Define Your Own List. Note That Punctuation, Like the Period Following the Word "if" Did Not Mess Up the Search for Lower Case (Nor Did the Quotation Marks Just Now). Punctuation Which is Screened Out of the Lower-Cased Word Search Pattern Include . , : ; ( ) [ ] ? ! ‘ ’ However, I Cannot Screen Text Braces; {For Example In} is Titled, Versus (for Example in), Since the Braces Are Not Screened Out in the Search for Pre-Designated Lower-Case Words Like for and in. However, `\textnc` Provides a Workaround: {for Example in}. Titlecap Will Consider Capitalizing Following a (, [, {, Or -Symbol, Such as (Abc-Def). You Can Use Your Text*xx* Commands, Like I Just Did Here with the Prior Xx, but if You Want the Argument of That Command to Not Be Titled, You Either Need, in This Example, to Add *Xx* to the Lowercase Word List, Which You Can See I Did Not. Instead, I Put "\relax xx" as the Argument, So That, in Essence, the \relax Was Capitalized, Not the X. Or You Could Use `\textnc` . Here I Demonstrate That Text Boldface, **as in the \textbf Command**, Also Works Fine, as Do `Texttt`, *Textsl*, TEXTSC, Textsf, *etc.* Titlecap Will Work on Diacritical Marks, Such as ŒApfel, ŒCacao *etc.*, Fontsize Changing Commands, as Well as National Symbols Such as Œlaf, Œgis, and Œdipus. Unfortunately, I Could Not Get It to Work on the å nor the ł symbols. the Method Will Work with Some Things in Math Mode, Capitalizing Symbols if There is a Leading Space, x^2 Can Become X^2, and It Can Process but It Will Not Capitalize the Greek Symbols, Such as α, and Will Choke on Most Macros, if They Are Not Direct Character Expansions. Additionally, Titlecaps Also Works with Font Changing Declarations, for Example, \itshape\sffamily. *You Can See That It Works Fine. Likewise, Any Subsequent \textxx Command Will, Upon Completion, Return the Font to Its Prior State, Such as This* **Textbf of Some Text**. *You Can See That I Have Returned to the Prior Font, Which Was Italic Sans-Serif. Now I Will Return to Upright Roman.* a Condition That Will Not Behave Well is Inner Braces, Such as `\titlecap{Blah {Inner Brace Material} Blah-Blah}`. See the Section on Quirks and Limitations for a Workaround Involving `\textnc`. Titlecap Will Always Capitalize the First Word of the Argument (**Even if It is on the Lower-Case Word List**), Unless `\titlecap` is Invoked with An Optional Argument That is Anything Other Than a Capital P. in That Case, the First Word Will Be Titled *Unless* It is on the Lowercase Word List. for Example, I Will Do a \titlecap[s]{a big man} and Get "a Big Man" with the "a" Not Titled. I Hope This Package is Useful to You, but as Far as Using Titlecaps on Such Large Paragraphs... **Do Not Try This At Home!**

6 Code Listing

```
\def\titlecapsVersionNumber{1.3}
\def\titlecapsVersionDate{2022/04/12}
\ProvidesPackage{titlecaps}
[\titlecapsVersionDate\ \titlecapsVersionNumber\
 Routines for setting rich-text input into Titling Caps]
%
% This work may be distributed and/or modified under the
% conditions of the LaTeX Project Public License, either version 1.3
% of this license or (at your option) any later version.
% The latest version of this license is in
%   http://www.latex-project.org/lppl.txt
% and version 1.3c or later is part of all distributions of LaTeX
% version 2005/12/01 or later.
%
% This work has the LPPL maintenance status `maintained'.
%
% The Current Maintainer of this work is Steven B. Segletes.
%
% V1.1  -Typographical corrections to docs.
%       -Missing % added on line 356
% V1.2  -Replaced all occurrences of \roman with \romannumeral\value
%       -Found two lines needing a trailing %
%       -Added a trailing space following invocations of \catcode
% V1.3  -Converted a number of \ifthenelse to corresponding TeX level
%        \ifx, \ifnum, and \if syntax
%       -Converted a number of \protected@edef to appropriately
%        expanded \def syntax.
%
\usepackage{ifnextok}
\usepackage{ifthen}
\newcounter{lcword@index}
\newcounter{word@count}
\newcounter{lc@words}
\let\SaveHardspace~
\def\SoftSpace{ }
\catcode`\^^00=12 %
\def\cmd@flag{^^00} % FLAGS END-OF-COMMAND; NEXT CHAR CAPPED
\def\def@x#1#2{\expandafter\def\expandafter#1\expandafter{#2}}
\def\def@xx#1#2{\expandafter\def@x\expandafter#1\expandafter{#2}}
\def\def@xxx#1#2{\expandafter\def@xx\expandafter#1\expandafter{#2}}

\let\sv@textup\textup
\let\sv@textit\textit
```

```
\let\sv@textsc\textsc
\let\sv@textsl\textsl
\let\sv@textmd\textmd
\let\sv@textbf\textbf
\let\sv@textrm\textrm
\let\sv@textsf\textsf
\let\sv@texttt\texttt
\let\sv@itshape\itshape
\let\sv@upshape\upshape
\let\sv@scshape\scshape
\let\sv@slshape\slshape
\let\sv@bfseries\bfseries
\let\sv@mdseries\mdseries
\let\sv@sffamily\sffamily
\let\sv@rmfamily\rmfamily
\let\sv@ttfamily\ttfamily

% THESE ARE THE PUNCTUATION MARKS SCREENED OUT FOR
% LOWER CASE WORD SEARCH
\newcommand\kill@punct{%
\catcode`.=9 %
\catcode`,=9 %
\catcode`:=9 %
\catcode`;=9 %
\catcode`(=9 %
\catcode`)=9 %
\catcode`[=9 %
\catcode`]=9 %
\catcode`?=9 %
\catcode`!=9 %
\catcode``=9 %
\catcode`'=9 %
}

\newcommand\restore@punct{%
\catcode`.=12 %
\catcode`,=12 %
\catcode`:=12 %
\catcode`;=12 %
\catcode`(=12 %
\catcode`)=12 %
\catcode`[=12 %
\catcode`]=12 %
\catcode`?=12 %
\catcode`!=12 %
\catcode``=12 %
```

```
\catcode‘’=12 %
}

\def\add@space{\def@xx\@thestring{\expandafter\@thestring\SoftSpace}}


% PRIMUS IS FOR BEGINNING-OF-STRING TITLE-CAPPING (1st WORD OVERRIDES
% PREDEFINED LOWER CASE)
\newcommand\redefine@primus{%
  \def\textup##1{\relax\bgroup\sv@upshape\titlecap[s]{{\cmd@flag##1}}\egroup}%
  \def\textit##1{\relax\bgroup\sv@itshape\titlecap[s]{{\cmd@flag##1}}\egroup}%
  \def\textsc##1{\relax\bgroup\sv@scshape\titlecap[s]{{\cmd@flag##1}}\egroup}%
  \def\textsl##1{\relax\bgroup\sv@slshape\titlecap[s]{{\cmd@flag##1}}\egroup}%
  \def\textmd##1{\relax\bgroup\sv@mdseries\titlecap[s]{{\cmd@flag##1}}\egroup}%
  \def\textbf##1{\relax\bgroup\sv@bfseries\titlecap[s]{{\cmd@flag##1}}\egroup}%
  \def\textrm##1{\relax\bgroup\sv@rmfamily\titlecap[s]{{\cmd@flag##1}}\egroup}%
  \def\textsf##1{\relax\bgroup\sv@sffamily\titlecap[s]{{\cmd@flag##1}}\egroup}%
  \def\texttt##1{\relax\bgroup\sv@ttfamily\titlecap[s]{{\cmd@flag##1}}\egroup}%
  \def\textnc##1{\relax\bgroup\titlecap[s]{{\cmd@flag##1}}\egroup}%
  \def\itshape{\relax\sv@itshape\cmd@flag}%
  \def\upshape{\relax\sv@upshape\cmd@flag}%
  \def\scshape{\relax\sv@scshape\cmd@flag}%
  \def\slshape{\relax\sv@slshape\cmd@flag}%
  \def\bfseries{\relax\sv@bfseries\cmd@flag}%
  \def\mdseries{\relax\sv@mdseries\cmd@flag}%
  \def\sffamily{\relax\sv@sffamily\cmd@flag}%
  \def\rmfamily{\relax\sv@rmfamily\cmd@flag}%
  \def\ttfamily{\relax\sv@ttfamily\cmd@flag}%
}
% SECONDUS IS FOR MIDSTRING TITLE-CAPPING (WHERE PREDEFINED LOWER
% CASE CAN PREVAIL)
\newcommand\redefine@secundus{%
  \def\textup##1{\relax\bgroup\sv@upshape\titlecap[s]{{##1}}\egroup}%
  \def\textit##1{\relax\bgroup\sv@itshape\titlecap[s]{{##1}}\egroup}%
  \def\textsc##1{\relax\bgroup\sv@scshape\titlecap[s]{{##1}}\egroup}%
  \def\textsl##1{\relax\bgroup\sv@slshape\titlecap[s]{{##1}}\egroup}%
  \def\textmd##1{\relax\bgroup\sv@mdseries\titlecap[s]{{##1}}\egroup}%
  \def\textbf##1{\relax\bgroup\sv@bfseries\titlecap[s]{{##1}}\egroup}%
  \def\textrm##1{\relax\bgroup\sv@rmfamily\titlecap[s]{{##1}}\egroup}%
  \def\textsf##1{\relax\bgroup\sv@sffamily\titlecap[s]{{##1}}\egroup}%
  \def\texttt##1{\relax\bgroup\sv@ttfamily\titlecap[s]{{##1}}\egroup}%
  \def\textnc##1{\relax\bgroup\titlecap[s]{{##1}}\egroup}%
  \def\itshape{\relax\sv@itshape\cmd@flag}%
  \def\upshape{\relax\sv@upshape\cmd@flag}%
  \def\scshape{\relax\sv@scshape\cmd@flag}%
  \def\slshape{\relax\sv@slshape\cmd@flag}%
```

```
  \def\bfseries{\relax\sv@bfseries\cmd@flag}%
  \def\mdseries{\relax\sv@mdseries\cmd@flag}%
  \def\sffamily{\relax\sv@sffamily\cmd@flag}%
  \def\rmfamily{\relax\sv@rmfamily\cmd@flag}%
  \def\ttfamily{\relax\sv@ttfamily\cmd@flag}%
}

% TERTIUS IS USED FOR STRIPPING OUT MACROS, SO THAT LOWER-CASE
% WORDS CAN BE FOUND
\newcommand\redefine@tertius{%
  \def\textup##1{\bgroup{##1}\egroup}%
  \def\textit##1{\bgroup{##1}\egroup}%
  \def\textsc##1{\bgroup{##1}\egroup}%
  \def\textsl##1{\bgroup{##1}\egroup}%
  \def\textmd##1{\bgroup{##1}\egroup}%
  \def\textbf##1{\bgroup{##1}\egroup}%
  \def\textrm##1{\bgroup{##1}\egroup}%
  \def\textsf##1{\bgroup{##1}\egroup}%
  \def\texttt##1{\bgroup{##1}\egroup}%
  \def\textnc##1{\bgroup{##1}\egroup}%
  \def\itshape{}%
  \def\itshape{}%
  \def\upshape{}%
  \def\scshape{}%
  \def\slshape{}%
  \def\bfseries{}%
  \def\mdseries{}%
  \def\sffamily{}%
  \def\rmfamily{}%
  \def\ttfamily{}%
}

\newcommand\un@define{%
  \let\textup\sv@textup%
  \let\textit\sv@textit%
  \let\textsc\sv@textsc%
  \let\textsl\sv@textsl%
  \let\textmd\sv@textmd%
  \let\textbf\sv@textbf%
  \let\textrm\sv@textrm%
  \let\textsf\sv@textsf%
  \let\texttt\sv@texttt%
  \def\textnc##1{##1}%
  \let\itshape\sv@itshape%
  \let\upshape\sv@upshape%
  \let\scshape\sv@scshape%
```

```
  \let\slshape\sv@slshape%
  \let\bfseries\sv@bfseries%
  \let\mdseries\sv@mdseries%
  \let\sffamily\sv@sffamily%
  \let\rmfamily\sv@rmfamily%
  \let\ttfamily\sv@ttfamily%
}

% USES EQUIVALENT NAMES FROM stringstrings PACKAGE
\newcommand\usestringstringsnames{%
  \let\addlcwords\Addlcwords%
  \let\resetlcwords\Resetlcwords%
  \let\addlcword\add@lcword%
  \let\getargs\get@argsC%
  \newcommand\capitalizetitle[2][v]{%
    \if v##1\titlecap[P]{##2}\else\titlecap@q[P]{##2}\fi%
  }%
}

% STORE (DON'T EXECUTE) \titlecap COMMAND & ARGUMENT
\newcommand\titlecap@q[2][P]{%
  \def\thestring{\titlecap[#1]{#2}}%
}

% RESET PREDESIGNATED LOWERCASE WORD LIST
\setcounter{lc@words}{0}
\newcommand\Resetlcwords[0]{\setcounter{lc@words}{0}}

% ADD WORDS TO PREDESIGNATED LOWERCASE WORD LIST
\newcommand\Addlcwords[1]{%
  \get@argsC{#1}%
  \setcounter{lcword@index}{0}%
  \whiledo{\value{lcword@index} < \narg}{%
    \addtocounter{lcword@index}{1}%
    \add@lcword{\csname arg\romannumeral\value{lcword@index}\endcsname}%
  }%
}

\newcommand\add@lcword[1]{%
  \addtocounter{lc@words}{1}%
  \expandafter\def@xx\csname lcword\romannumeral\value{lc@words}\endcsname{#1}%
}

% SEARCH TERTIUS CONVERTED ARGUMENT FOR LOWERCASE WORDS, SET FLAG
% FOR EACH WORD (T = FOUND IN LIST, F= NOT FOUND IN LIST)
\newcommand\seek@lcwords[1]{%
```

```
\kill@punct%
  \setcounter{word@count}{0}%
  \whiledo{\value{word@count} < \narg}{%
    \addtocounter{word@count}{1}%
    \def@xx\current@word{%
      \csname arg\romannumeral\value{word@count}\endcsname}%
    \def\found@word{F}%
    \setcounter{lcword@index}{0}%
    \expandafter\def\csname%
            found@word\romannumeral\value{word@count}\endcsname{F}%
    \whiledo{\value{lcword@index} < \value{lc@words}}{%
      \addtocounter{lcword@index}{1}%
      \def@xx\current@lcword{%
        \csname lcword\romannumeral\value{lcword@index}\endcsname}%
%% THE FOLLOWING THREE LINES ARE FROM DAVID CARLISLE
  \protected@edef\tmp{\noexpand\scantokens{\def\noexpand\tmp%
   {\noexpand\ifthenelse{\noexpand\equal{\current@word}{\current@lcword}}}}}%
  \tmp\ifhmode\unskip\fi\tmp
%%
      {\expandafter\def\csname%
            found@word\romannumeral\value{word@count}\endcsname{T}%
       \setcounter{lcword@index}{\value{lc@words}}}%
      {}%
    }%
  }%
\if P#1\def\found@wordi{F}\fi%
\restore@punct%
}

% THE TITLECAP ROUTINE
\newcommand\titlecap[2][P]{%
  \digest@sizes%
  \if T\converttilde\def~{ }\fi%
  \redefine@tertius%
  \get@argsC{#2}%
  \seek@lcwords{#1}%
  \if P#1%
    \redefine@primus%
    \get@argsC{#2}%
    \def@x\primus@argi{\argi}%
  \else%
  \fi%
  \setcounter{word@count}{0}%
  \redefine@secundus%
  \def\@thestring{}%
  \get@argsC{#2}%
```

```
  \if P#1\def@x\argi{\primus@argi}\fi%
  \whiledo{\value{word@count} < \narg}{%
    \addtocounter{word@count}{1}%
    \if F\csname found@word\romannumeral\value{word@count}\endcsname%
      \title@word{%
        \csname arg\romannumeral\value{word@count}\endcsname}%
      \expandafter\def@x\csname%
           arg\romannumeral\value{word@count}\endcsname{\@thestring}%
    \else%
      \notitle@word{%
        \csname arg\romannumeral\value{word@count}\endcsname}%
      \expandafter\def@x\csname%
           arg\romannumeral\value{word@count}\endcsname{\@thestring}%
    \fi%
  }%
  \def\@thestring{}%
  \setcounter{word@count}{0}%
  \whiledo{\value{word@count} < \narg}{%
    \addtocounter{word@count}{1}%
    \ifnum\value{word@count} = 1\relax\else\add@space\fi%
    \def@xxx\@thestring{\expandafter\expandafter\expandafter\@thestring%
      \csname arg\romannumeral\value{word@count}\endcsname}%
  }%
  \let~\SaveHardspace%
  \@thestring%
  \restore@sizes%
\un@define}

\newcommand\notitle@word[1]{%
  \def\symbol@flag{F}%
  \def@xx\the@string{#1}%
  \def\@thestring{}\def\make@cap{F}%
  \expandafter\eat@noTitleWord\the@string\string@end%
}

\def\eat@noTitleWord{\def\make@cap{F}\IfNextToken\string@end%
  {\@gobble}%
  {\title@string{\eat@noTitleWord}}%
}

\newcommand\title@word[1]{%
  \def\symbol@flag{F}%
  \def@xx\the@string{#1}%
  \def\@thestring{}\def\make@cap{T}%
  \expandafter\eat@TitleWord\the@string\string@end%
}
```

```
\def\eat@TitleWord{\IfNextToken\string@end%
  {\@gobble}%
  {\title@string{\eat@TitleWord}}%
}

\def\@symboli{\‘}
\def\@symbolii{\’}
\def\@symboliii{\^}
\def\@symboliv{\"}
\def\@symbolv{\~}
\def\@symbolvi{\=}
\def\@symbolvii{\.}
\def\@symbolviii{\u}
\def\@symbolix{\v}
\def\@symbolx{\H}
\def\@symbolxi{\t}
\def\@symbolxii{\c}
\def\@symbolxiii{\d}
\def\@symbolxiv{\b}
\def\@symbolxv{\oe}
 \def\uc@symbolxv{\OE}
\def\@symbolxvi{\ae}
 \def\uc@symbolxvi{\AE}
\def\@symbolxvii{\o}
 \def\uc@symbolxvii{\O}
%\def\@symbolxviii{\aa}
% \def\uc@symbolxviii{\AA}
%\def\@symbolxix{\l}
% \def\uc@symbolxix{\L}
\newcounter{dia@count}

\def\title@string#1#2{%
  \if T\make@cap%
    \setcounter{dia@count}{1}%
    \if F\symbol@flag%
      \whiledo{\value{dia@count} < 18}{%
        \expandafter\expandafter\expandafter\ifx
           \csname @symbol\romannumeral\value{dia@count}\endcsname#2%
%BEGIN IFDIACRIT
          \ifnum\value{dia@count} < 15\relax%
%IF = DIACRIT<15
            \def@xx\di@critic%
              {\csname @symbol\romannumeral\value{dia@count}\endcsname}%
            \def\symbol@flag{D}%
```

```
            \setcounter{dia@count}{99}% INDICATING DICRIT JUST FOUND
           \else%
%IF = NATSYM
            \def@xx\di@critic%
              {\csname uc@symbol\romannumeral\value{dia@count}\endcsname}%
            \setcounter{dia@count}{90}% >19 AND <99 MEANS NON-DIACRIT SYMBOL
            \def\symbol@flag{N}%
           \fi%
         \else
%END IF = DIACRIT
%IFNOT = DIACRIT
          \addtocounter{dia@count}{1}\fi
      }% END WHILEDO
    \fi%
    \ifnum\value{dia@count} < 99\relax
      \if D\symbol@flag% FOR DIACRIT, ONCE ARGUMENT IS IN #2, TO BE CAPPED
        \def\next@char{\di@critic#2}%
        \def\symbol@flag{F}%
      \else%
        \if N\symbol@flag% FOR NATSYM TO BE CAPITALIZED
          \def\next@char{\di@critic}%
          \def\symbol@flag{F}%
        \else% FOR ANY OTHER CHARACTER TO BE CAPITALIZED
          \def\next@char{#2}%
        \fi%
      \fi%
      \expandafter\ifx\cmd@flag#2\relax\def\make@cap{T}\else%
        \def@xx\@thestring{\expandafter\@thestring\expandafter
          \uppercase\expandafter{\next@char}}%
        \def\make@cap{F}%
        \@checkfornewgroup{#2}%
      \fi
    \fi%
  \else% FOR CHARACTERS NOT TO BE CAPITALIZED
    \expandafter\ifx\cmd@flag#2
      \def\make@cap{T}%
    \else
      \def@x\@thestring{\@thestring#2}%
      \@checkfornewgroup{#2}%
    \fi
  \fi%
  #1}

\def\@checkfornewgroup#1{%
  \ifx-#1\def\make@cap{T}\else
    \ifx(#1\def\make@cap{T}\else
```

```
      \ifx[#1\def\make@cap{T}\else
        \ifx\{#1\def\make@cap{T}\else
          \ifx‘#1\def\make@cap{T}\fi
        \fi
      \fi
    \fi
  \fi
}

%%%%%%%%%%%%%%%%%%
% DAVID CARLISLE GREATLY ASSISTED WITH THE \get@argsC COMMAND LOGIC

\def\string@@@end{$\SaveHardspace}
\def\converttilde{F}
\newcounter{arg@@@index}
\let\SaveHardspace~%%%
\def\the@@@rule{\rule{.8ex}{1.6ex}}%

\def\get@argsC#1{%
  \if T\converttilde\def~{ }\else\catcode‘~=12 \fi
  \protected@edef\the@@@string{#1}%
  \setcounter{arg@@@index}{0}%
  \lowercase{\expandafter\parse@@@Block\the@@@string} \string@@@end
  \let~\SaveHardspace%
  \catcode‘~=13 %
}

\def\parse@@@Block#1 {%
  \stepcounter{arg@@@index}%
  \@namedef{arg\romannumeral\value{arg@@@index}}{#1}%
  \ifthenelse{\equal{\argi}{}}{\addtocounter{arg@@@index}{-1}}{}%
  \futurelet\tmp\parse@@@Block@@@}

\def\parse@@@Block@@@{%
\ifx\tmp\string@@@end\edef\narg{\thearg@@@index}\expandafter\@gobble%
\else\expandafter\parse@@@Block\fi}

%%%%%%%%%%%

\let\sv@tiny\tiny
\let\sv@scriptsize\scriptsize
\let\sv@footnotesize\footnotesize
\let\sv@small\small
\let\sv@normalsize\normalsize
\let\sv@large\large
\let\sv@Large\Large
```

```
\let\sv@LARGE\LARGE
\let\sv@huge\huge
\let\sv@huge\Huge

\let\make@lttr\makeatletter%
\let\make@othr\makeatother%

\newcommand\noatinsidetc{%
  \def\make@lttr{}%
  \def\make@othr{}%
}

\def\restore@sizes{%
  \let\tiny\sv@tiny%
  \let\scriptsize\sv@scriptsize%
  \let\footnotesize\sv@footnotesize%
  \let\small\sv@small%
  \let\normalsize\sv@normalsize%
  \let\large\sv@large%
  \let\Large\sv@Large%
  \let\LARGE\sv@LARGE%
  \let\huge\sv@huge%
  \let\huge\sv@Huge%
  \make@othr%
}

% THE \makeatletter IS REQUIRED FOR PROCESSING FONTSIZE CHANGES
\def\digest@sizes{%
\make@lttr%
\def\tiny{\unskip\sz@tiny\SoftSpace}%
\def\sz@tiny{\sv@tiny}%
%
\def\scriptsize{\unskip\sz@scriptsize\SoftSpace}%
\def\sz@scriptsize{\sv@scriptsize}%
%
\def\footnotesize{\unskip\sz@footnotesize\SoftSpace}%
\def\sz@footnotesize{\sv@footnotesize}%
%
\def\small{\unskip\sz@small\SoftSpace}%
\def\sz@small{\sv@small}%
%
\def\normalsize{\unskip\sz@normalsize\SoftSpace}%
\def\sz@normalsize{\sv@normalsize}%
%
\def\large{\unskip\sz@large\SoftSpace}%
\def\sz@large{\sv@large}%
```

```
%
\def\Large{\unskip\sz@Large\SoftSpace}%
\def\sz@Large{\sv@Large}%
%
\def\LARGE{\unskip\sz@LARGE\SoftSpace}%
\def\sz@LARGE{\sv@LARGE}%
%
\def\huge{\unskip\sz@huge\SoftSpace}%
\def\sz@huge{\sv@huge}%
%
\def\Huge{\unskip\sz@Huge\SoftSpace}%
\def\sz@Huge{\sv@Huge}%
}

\endinput
```