The cmdtrack package

Michael Downes

Version 1.05, 1999/07/22

1 Introduction

The cmdtrack package aids in the task of checking whether a command defined in a document preamble is actually used somewhere in the document. If you add a statement

```
\usepackage{cmdtrack}
```

in the preamble of your document, all \newcommand and similar statements between that point and \begin{document} will be marked for logging. At the end of the document a report of the command usage will be printed in the TEX log, for example:

```
"mdash" was used on line 25
"ndash" was never used
"gar " was used on line 26
"math character ?" was used on line 26
```

In this list, a command \foo is normally listed as "foo"; however if it was defined with an optional argument, it will be listed as "\foo", and if defined with \DeclareRobustCommand it will be listed as "foo␣" (with an extra space). This has to do with the way LATEX deals with such definitions behind the scenes. Twiddling the behavior to get more consistent treatment of the command names looked hard, so I didn't do it. If you can send actual code to remedy this deficiency, it will be welcome. I do not plan on trying to fix it myself until after I have dealt with all the other, more interesting or more pressing, problems in my queue.

Certain kinds of commands are inherently untrackable due to the way they are used (counters, lengths, and other variables that may appear on the right-hand of an assignment statement; commands whose first use is in the argument of \label, and so on.) Tracking for commands of this sort can be turned off by moving their definitions earlier (before the load statement of the cmdtrack package), or by listing them in the argument of an \untrack statement at the end of your document preamble, for example:

```
\untrack{\foo,\bar,\blub,...}
```

2 Limitations

- Only commands defined with `\newcommand`, `\newenvironment`, `\newtheorem`, `\DeclareMathSymbol`, `\DeclareMathOperator`, and their variants are logged. Commands will not be logged if they are defined with `\def` instead of `\newcommand`, `\mathchardef` instead of `\DeclareMathSymbol`, etc.
- "Commands" defined with `\newlength`, `\newsavebox`, `\newcounter`, `\newcount`, `\newtoks`, etc., will not be logged because it cannot be done without disrupting their normal use.
- Commands defined with `\DeclareTextSymbol`, `\DeclareTextAccent`, `\DeclareMathRadical`, `\DeclareMathAlphabet`, and various other things will not be logged because I was starting to doze off on the keyboard after adding the support for `\DeclareMathSymbol`. Any volunteers?
- The definitions of the commands that are to be logged must fall between `\usepackage{cmdtrack}` and `\begin{document}`. For problem commands, try moving their definitions before the `\usepackage{cmdtrack}` line, or use the `\untrack` command to cancel tracking for selected commands at the end of the preamble.
- The `cmdtrack` package should be the last package loaded (or, the later-loaded packages will have their commands tracked as well, if that is what you want).
- A command definition starting with `\multicolumn` will cause trouble (unless it is never used).
- Large number of `\newcommands` (200+, say) might lead to an error message "TeX capacity exceeded (hash size ...)". Depends on the capacity of your TeX system.

3 Embedded documentation

The following material comes into play only when the `?` package option is used.

```
%% Self-documenting section
\ifcat ?$\relax{\catcode37=7 \catcode127=9 \def\0{\@sanitize\catcode}\fi
%%? \endlinechar125\catcode127=13\def%%?{\typeout}\037=7
%%?{==================================================================
%%?{ With the cmdtrack package, all commands and environments defined
%%?{ between \usepackage{cmdtrack} and \begin{document} will be marked
%%?{ for logging. A report on the usage of the marked commands will be
%%?{ printed in the LaTeX log file. Standard LaTeX methods must be used
%%?{ for defining the commands (things defined with \def, for example,
%%?{ won't be logged). Use \untrack{\cmd,\othercmd,...} just before
%%?{ \begin{document} to turn off tracking for selected commands.
%%?{
%%?{ Package options:
%%?{
%%?{ ?       Causes this information to be shown on-screen.
%%?{
%%?{ morose  Opposite of verbose: causes the brief message about the ?
```

```
%%?{        option to be suppressed.
%%?{
%%?{ Support for the following is not [yet] provided:
%%?{ \DeclareTextSymbol, \DeclareMathRadical, \DeclareMathAlphabet, and
%%?{ some others.
%%?{
%%?{ More comprehensive documentation for cmdtrack.sty may be found in
%%?{ cmdtrack.dtx.
%%?{===============================================================
%%?{}}\endinput\bgroup
%%
```

4 Implementation

Standard declaration of package name and date.

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{cmdtrack}[1999/07/22 v1.05]
```

The `cmdtrack` package works by hooking into the internal LaTeX function `\@yargdef`. When processing something like `\newcommand{\foo}{...}`, there is a point where LaTeX runs, essentially,

```
\def\foo{...}
```

Just as LaTeX is about to carry out that assignment, we jump in and substitute a different control sequence in place of `\foo`! Our substituted control sequence has the same name but with an extra double-quote character `"` added at the beginning: i.e., `\"foo` instead of `\foo`. Thus `\"foo` gets the real definition, and we are free to give `\foo` some other definition for logging purposes. Obviously it should end by calling the real definition. And here is how we arrange that.

1. `\foo` expands to `\logcmd \"foo \foo`.
2. `\logcmd` redefines `\foo` to take on the meaning of `\"foo`.
3. `\logcmd` adds `\"foo` to a used-commands list, and records the current line number with it.

Then when we reach the end of our document, we need to find some way of looking through all of our marked commands and reporting which ones were used. During the processing of the preamble, the marked commands were stored in a list `\commandlist`. Furthermore, when `\logcmd` redefined `\"foo`, it put in a special marker character along with the line number. Then when we reach the end of our document, we can iterate over the command list with a function that checks for that special character in the `\meaning` of each command.

Alternatively one could imagine adding the used commands to a separate list. At one point I started doing that but it got a little messy so I abandoned the idea. Besides, this way the report comes out with the commands in the same order as they were defined in the preamble (except that newtheorem commands gravitate toward the end of the list).

```
\let\commandlist\@empty
```

```
\AtEndDocument{\report@command@usage}
```

The LaTeX kernel doesn't provide this.

```
\edef\@quotechar{\string"}
```

The `\untrack` command could just redefine `\logcmd` and run through its argument list, but if the user accidentally leaves in some command that is no longer defined (or defined before the `cmdtrack` package is loaded) then some kind of error is likely. So we check specifically for the string `\logcmd` at the beginning of each command's `\meaning` string.

```
\newcommand{\untrack}[1]{%
  \begingroup
  \def\logcmd##1##2{\global\let##1=##2%
    \xdef##2{\@percentchar\the\inputlineno}%
  }%
  \untrack@a#1{,\@gobble}\endgroup
}
```

```
i
\def\untrack@a#1{%
  \ifx,#1\@gobbletwo\expandafter\@gobble
  \else \expandafter\untrack@b\meaning#1\@nil#1%
  \fi
  \untrack@a
}
```

```
\def\untrack@b#1->#2#3#4#5#6#7#8#9\@nil{%
  \expandafter\ifx\csname #3#4#5@#6#7#8\endcsname\log@cmd
  \else
    \def\@tempa##1{%
      \PackageWarningNoLine{cmdtrack}{%
Command \protect##1 does not have tracking turned on}%
    }%
    \expandafter\@tempa
  \fi
}
```

One branch of providecommand processing uses `\reserved@a` in a way that will fail unless we watch out for it.

```
\def\@isreserved@a#1\reserved@a#2#3@{#2}
```

```
\let\@hash@\relax
\def\log@cmd#1#2{%
  \if\@isreserved@a#2T\reserved@a F@T%
    \endgroup
  \else
    \toks@\expandafter{\commandlist#1}%
    \xdef\commandlist{\the\toks@}%
    \endgroup
    \def#2{\logcmd#2#1}%
  \fi
  \let\@hash@##%
```

```
    \l@ngrel@x\expandafter\def\expandafter#1\reserved@a
  }
  \def\logcmd#1#2{%
    \ifx\protect\@typeset@protect
      \global\let#1#2%
      \xdef#2{\@percentchar\the\inputlineno}%
    \else
      \expandafter\protect
    \fi
    #1%
  }
  \begingroup \catcode`\"=12
  \gdef\cmd@check#1->#2#3-#4\@nil#5{%
    \if\@percentchar#2\typeout{\string#5" was used on line #3}%
    \else\typeout{\string#5" was never used}%
    \fi
  }
  \endgroup
  \def\report@command@usage{%
    \def\@tempa{\typeout{========================================}}%
    \@tempa
    \begingroup \escapechar\m@ne
    \def\do##1{%
      \ifx\advance##1\expandafter\@gobbletwo\fi
      {\expandafter\cmd@check\meaning##1->@-\@nil##1}%
      \do
    }%
    \expandafter\do\commandlist \advance\z@\z@
    \endgroup
    \@tempa
  }
  \def\testthm#1{%
    \expandafter\testthm@a\csname#1\expandafter\endcsname
      \csname\@quotechar#1\endcsname
  }
```

In order to avoid figuring out how to read the optional arguments of `\newtheorem`, we postpone the application of `\log@cmd` (actually, a slight variation thereof) until `\begin{document}`.

```
  \let\old@newtheorem\newtheorem
  \def\newtheorem#1{%
    \AtBeginDocument{\testthm{#1}}%
    \old@newtheorem{#1}%
  }
  \def\testthm@a#1#2{%
    \begingroup
    \toks@\expandafter{\commandlist#2}%
    \xdef\commandlist{\the\toks@}%
  \endgroup
  \let#2#1%
  \def#1{\logcmd#1#2}%
}
```

It would be unusual for a character to be defined as a math symbol in a document preamble. But what hey, why not support it?

```
\def\set@mathchar#1#2#3#4{%
  \expandafter\set@mathchar@a
  \csname\@quotechar math character \string#2\expandafter\endcsname
  \expandafter{\number`#2}{#1}{#3}{#4}%
}%

\def\set@mathchar@a#1#2#3#4#5{%
  \global\mathcode#2=\@quotechar 8000
  \global\mathchardef#1\@quotechar\mathchar@type#4\hexnumber@#3#5\relax
  \toks@\expandafter{\commandlist#1}%
  \xdef\commandlist{\the\toks@}%
  \begingroup \lccode`\.=#2\lccode`\~=#2\lowercase{\endgroup
    \gdef~{\global\mathcode#2=#1\logcmd#1#1}}%
}

\def\set@mathsymbol#1#2#3#4{%
  \begingroup \escapechar=`\"\relax
  \global\expandafter\mathchardef
    \csname\string#2\endcsname
    \@quotechar\mathchar@type#3\hexnumber@#1#4\relax
  \expandafter\log@cmd@a\csname\string#2\endcsname#2%
}

\def\set@mathaccent#1#2#3#4{%
  \xdef#2{\mathaccent\@quotechar\mathchar@type#3\hexnumber@#1#4\relax}%
  \begingroup \escapechar`\"\relax
  \expandafter\log@cmd@a\csname\string#2\endcsname#2%
}
```

Note the use of `\gdef`.

```
\def\log@cmd@a#1#2{%
  \toks@\expandafter{\commandlist#1}%
  \xdef\commandlist{\the\toks@}%
  \endgroup
  \gdef#2{\logcmd#2#1}%
}

\DeclareOption{?}{\AtEndOfPackage{\ShowPackageInfo{cmdtrack}}}

\DeclareOption{morose}{}

\DeclareOption{simple}{%
  \def\logcmd#1#2{%
    \ifx\protect\@typeset@protect
      \global\let#1#2%
      \begingroup \escapechar\m@ne
      \typeout{\string#2\string" was used on line \the\inputlineno}%
      \endgroup
    \else
      \expandafter\protect
    \fi
    #1%
  }%
  \AtBeginDocument{\global\let\commandlist\@empty}%
  \global\let\report@command@usage\relax
}
```

```
\ProcessOptions\relax
\begingroup
\catcode`\%=9 \catcode`\&=14 \catcode`\"=12
\@ifpackagewith{cmdtrack}{morose}{\catcode`\%=14 }{}
\@ifpackagewith{cmdtrack}{?}{\catcode`\%=14 }{}
%%\typeout{&
%%Try "\string\usepackage[\string ?]{cmdtrack}"
%%to see information on using this package^^J&
%%[including how to turn off this "helpful" message].&
%%}
\endgroup
```

```
\newcommand{\ShowPackageInfo}[1]{%
  \begingroup \catcode`\?=3
  \input{#1.\@pkgextension}%
  \endgroup
}
```

```
\let\@@yargdef\@yargdef
```

Restore yargdef to normal at beginning of document, so that uses of `\newcommand` in `\maketitle` (e.g.) are ignored.

```
\AtBeginDocument{\let\@yargdef\@@yargdef}
```

```
\let\@hash@\relax
\def\@yargdef #1#2#3{%
  \@tempcnta#3\relax \advance\@tempcnta\@ne \let\@hash@\relax
  \edef\reserved@a{\ifx#2\tw@ [\@hash@ 1]\fi}%
  \@tempcntb#2%
  \@whilenum\@tempcntb<\@tempcnta\do{%
    \edef\reserved@a{\reserved@a\@hash@\the\@tempcntb}%
    \advance\@tempcntb\@ne
  }%
  \begingroup \escapechar=`\"\relax
  \expandafter\log@cmd\csname\string#1\endcsname#1%
}
```

The usual `\endinput` to ensure that random garbage at the end of the file doesn't get copied by `docstrip`.

```
\endinput
```