

The `mdwtools.tex`* file

Mark Wooding

10 May 1996

Contents

1	Introduction and user guide	1	2.8	Float handling	7
1.1	Declarations	1	2.9	Other bits of parameter tweaking	7
1.1.1	Describing packages	2	2.10	Contents handling	8
1.1.2	Contents handling	2	2.11	Finishing it all off	8
1.1.3	Other declarations	2	2.12	File version information .	9
1.2	Other commands	2	2.13	List handling	10
1.3	Customisation	2	2.14	Described file handling . .	10
2	Implementation	3	2.15	Author and title handling	11
2.1	Initialisation	3	2.16	Building title strings . . .	12
2.2	Some macros for interac- tion	4	2.17	Starting the main docu- ment	13
2.3	Decide on indexing	4	2.18	And finally...	14
2.4	Selectively defining things	6			
2.5	General markup things . .	6			
2.6	A table environment . . .	6	A	The GNU General Public License	15
2.7	Commenting out of stuff .	7			

1 Introduction and user guide

This file is really rather strange; it gets `\input` by other package documentation files to set up most of the environmental gubbins for them. It handles almost everything, like loading a document class, finding any packages, and building and formatting the title.

It also offers an opportunity for users to customise my nice documentation, by using a `mdwtools.cfg` file (not included).

1.1 Declarations

A typical documentation file contains something like

```
\input{mdwtools}
<declarations>
\mdwdoc
```

*The `mdwtools.tex` file is currently at version 1.4, dated 10 May 1996.

The initial `\input` reads in this file and sets up the various commands which may be needed. The final `\mdwdoc` actually starts the document, inserting a title (which is automatically generated), a table of contents etc., and reads the documentation file in (using the `\DocInput` command from the `doc` package).

1.1.1 Describing packages

<code>\describepackage</code> <code>\describesclass</code> <code>\describesfile</code> <code>\describesfile*</code>	<p>The most important declarations are those which declare what the documentation describes. Saying <code>\describepackage{<package>}</code> loads the <i><package></i> (if necessary) and adds it to the auto-generated title, along with a footnote containing version information. Similarly, <code>\describesclass</code> adds a document class name to the title (without loading it – the document itself must do this, with the <code>\documentclass</code> command). For files which aren't packages or classes, use the <code>\describesfile</code> or <code>\describesfile*</code> command (the *-version won't <code>\input</code> the file, which is handy for files like <code>mdwtools.tex</code>, which are already input).</p>
<code>\author</code> <code>\date</code> <code>\title</code>	<p>The <code>\author</code>, <code>\date</code> and <code>\title</code> declarations work slightly differently to normal – they ensure that only the <i>first</i> declaration has an effect. (Don't you play with <code>\author</code>, please, unless you're using this program to document your own packages.) Using <code>\title</code> suppresses the automatic title generation.</p>
<code>\docdate</code>	<p>The default date is worked out from the version string of the package or document class whose name is the same as that of the documentation file. You can choose a different 'main' file by saying <code>\docdate{<file>}</code>.</p>

1.1.2 Contents handling

<code>\addcontents</code>	<p>A documentation file always has a table of contents. Other contents-like lists can be added by saying <code>\addcontents{<extension>}{<command>}</code>. The <i><extension></i> is the file extension of the contents file (e.g., 'lot' for the list of tables); the <i><command></i> is the command to actually typeset the contents file (e.g., <code>\listoftables</code>).</p>
---------------------------	---

1.1.3 Other declarations

<code>\implementation</code>	<p>The <code>doc</code> package wants you to say <code>\StopEventually{<stuff>}</code> before describing the package implementation. Using <code>mdwtools.tex</code>, you just say <code>\implementation</code>, and everything works. It will automatically read in the licence text (from <code>gpl.tex</code>, and wraps some other things up.</p>
------------------------------	---

1.2 Other commands

The `mdwtools.tex` file includes the `syntax` and `sverb` packages so that they can be used in documentation files. It also defines some trivial commands of its own.

<code>\<</code> <code>\smallf</code>	<p>Saying <code>\<{<text>}</code> is the same as <code>\synt{<text>}</code>; this is a simple abbreviation.</p> <p>Saying <code>\smallf <number>/<number></code> typesets a little fraction, like this: $\frac{3}{4}$. It's useful when you want to say that the default value of a length is 2 1/2 pt, or something like that.</p>
--	---

1.3 Customisation

You can customise the way that the package documentation looks by writing a file called `mdwtools.cfg`. You can redefine various commands (before they're defined

here, even; `mdwtools.tex` checks most of the commands that it defines to make sure they haven't been defined already.

`\indexing` If you don't want the prompt about whether to generate index files, you can define the `\indexing` command to either 'y' or 'n'. I'd recommend that you use `\providecommand` for this, to allow further customisation from the command line.

`\mdwdateformat` If you don't like my date format (maybe you're American or something), you can redefine the `\mdwdateformat` command. It takes three arguments: the year, month and date, as numbers; it should expand to something which typesets the date nicely. The default format gives something like '10 May 1996'. You can produce something rather more exotic, like '10th May MCMXCVI' by saying

```
\newcommand{\mdwdateformat}[3]{%
  \number#3\textsuperscript{\numsuffix{#3}}\ %
  \monthname{#2}\ %
  \textsc{\romannumeral #1}%
}
```

`\monthname` Saying `\monthname{<number>}` expands to the name of the numbered month (which can be useful when doing date formats). Saying `\numsuffix{<number>}` will expand to the appropriate suffix ('th' or 'rd' or whatever) for the `<number>`. You'll have to superscript it yourself, if this is what you want to do. Putting the year number in roman numerals is just pretentious ;-).

`\mdwhook` After all the declarations in `mdwtools.tex`, the command `\mdwhook` is executed, if it exists. This can be set up by the configuration file to do whatever you want.

There are lots of other things you can play with; you should look at the implementation section to see what's possible.

2 Implementation

1 `(*mdwtools)`

The first thing is that I'm not a L^AT_EX package or anything official like that, so I must enable '@' as a letter by hand.

2 `\makeatletter`

Now input the user's configuration file, if it exists. This is fairly simple stuff.

3 `\@input{mdwtools.cfg}`

Well, that's the easy bit done.

2.1 Initialisation

Obviously the first thing to do is to obtain a document class. Obviously, it would be silly to do this if a document class has already been loaded, either by the package documentation or by the configuration file.

The only way I can think of for finding out if a document class is already loaded is by seeing if the `\documentclass` command has been redefined to raise an error. This isn't too hard, really.

```
4 \ifx\documentclass\@twoclasseserror\else
5   \documentclass[a4paper]{ltxdoc}
6   \ifx\doneclasses\mdw@undefined\else\doneclasses\fi
7 \fi
```

As part of my standard environment, I'll load some of my more useful packages. If they're already loaded (possibly with different options), I'll not try to load them again.

```
8 \ifpackage{doc}{\usepackage{doc}}
9 \ifpackage{syntax}{\usepackage[rounded]{syntax}}
10 \ifpackage{sverb}{\usepackage{sverb}}
```

2.2 Some macros for interaction

I like the L^AT_EX star-boxes, although it's a pain having to cope with T_EX's space-handling rules. I'll define a new typing-out macro which makes spaces more significant, and has a *-version which doesn't put a newline on the end, and interacts prettily with `\read`.

First of all, I need to make spaces active, so I can define things about active spaces.

```
11 \begingroup\obeyspaces
```

Now to define the main macro. This is easy stuff. Spaces must be carefully rationed here, though.

I'll start a group, make spaces active, and make spaces expand to ordinary space-like spaces. Then I'll look for a star, and pass either `\message` (which doesn't start a newline, and interacts with `\read` well) or `\immediate\write 16` which does a normal write well.

```
12 \gdef\mdwtype{%
13 \begingroup\catcode'\ \active\let \space%
14 \ifstar{\mdwtype@i{\message}}{\mdwtype@i{\immediate\write\sixt@@n}}%
15 }
16 \endgroup
```

Now for the easy bit. I have the thing to do, and the thing to do it to, so do that and end the group.

```
17 \def\mdwtype@i#1#2{#1{#2}\endgroup}
```

2.3 Decide on indexing

A configuration file can decide on indexing by defining the `\indexing` macro to either 'y' or 'n'. If it's not set, then I'll prompt the user.

First of all, I want a switch to say whether I'm indexing.

```
18 \newif\ifcreateindex
```

Right: now I need to decide how to make progress. If the macro's not set, then I want to set it, and start a row of stars.

```
19 \ifx\indexing\@undefined
20 \mdwtype{*****}
21 \def\indexing{?}
22 \fi
```

Now enter a loop, asking the user whether to do indexing, until I get a sensible answer.

```
23 \loop
24 \@tempswafalse
```

```

25 \if y\indexing\@tempwattrue\createindextrue\fi
26 \if Y\indexing\@tempwattrue\createindextrue\fi
27 \if n\indexing\@tempwattrue\createindexfalse\fi
28 \if N\indexing\@tempwattrue\createindexfalse\fi
29 \if@tempwa\else
30 \mdwtype{* Create index files? (y/n) *}
31 \read\sixt@@n to\indexing%
32 \repeat

```

Now, based on the results of that, display a message about the indexing.

```

33 \mdwtype{*****}
34 \ifcreateindex
35 \mdwtype{* Creating index files *}
36 \mdwtype{* This may take some time *}
37 \else
38 \mdwtype{* Not creating index files *}
39 \fi
40 \mdwtype{*****}

```

Now I can play with the indexing commands of the doc package to do whatever it is that the user wants.

```

41 \ifcreateindex
42 \CodelineIndex
43 \EnableCrossrefs
44 \else
45 \CodelineNumbered
46 \DisableCrossrefs
47 \fi

```

And register lots of plain T_EX things which shouldn't be indexed. This contains lots of \if... things which don't fit nicely in conditionals, which is a shame. Still, it doesn't matter that much, really.

```

48 \DoNotIndex{\def,\long,\edef,\xdef,\gdef,\let,\global}
49 \DoNotIndex{\if,\ifnum,\ifdim,\ifcat,\ifmmode,\ifvmode,\ifhmode,%
50 \iftrue,\iffalse,\ifvoid,\ifx,\ifeof,\ifcase,\else,\or,\fi}
51 \DoNotIndex{\box,\copy,\setbox,\unvbox,\unhbox,\hbox,%
52 \vbox,\vtop,\vcenter}
53 \DoNotIndex{\@empty,\immediate,\write}
54 \DoNotIndex{\egroup,\bgroup,\expandafter,\begingroup,\endgroup}
55 \DoNotIndex{\divide,\advance,\multiply,\count,\dimen}
56 \DoNotIndex{\relax,\space,\string}
57 \DoNotIndex{\csname,\endcsname,\@spaces,\openin,\openout,%
58 \closein,\closeout}
59 \DoNotIndex{\catcode,\endinput}
60 \DoNotIndex{\jobname,\message,\read,\the,\m@ne,\noexpand}
61 \DoNotIndex{\hsize,\vsize,\hskip,\vskip,\kern,\hfil,\hfill,\hss}
62 \DoNotIndex{\m@ne,\z@,\z@skip,\@ne,\tw@,\p@}
63 \DoNotIndex{\dp,\wd,\ht,\vss,\unskip}

```

Last bit of indexing stuff, for now: I'll typeset the index in two columns (the default is three, which makes them too narrow for my tastes).

```

64 \setcounter{IndexColumns}{2}

```

2.4 Selectively defining things

I don't want to tread on anyone's toes if they redefine any of these commands and things in a configuration file. The following definitions are fairly evil, but should do the job OK.

`\@gobbledef` This macro eats the following `\definition`, leaving not a trace behind.

```
65 \def\@gobbledef#1#\@gobble}
```

`\tdef` The `\tdef` command is a sort of 'tentative' definition – it's like `\def` if the control sequence named doesn't already have a definition. `\tlet` does the same thing with `\let`.

```
66 \def\tdef#1{
67   \ifx#1\@undefined%
68     \expandafter\def\expandafter#1%
69   \else%
70     \expandafter\@gobbledef%
71   \fi%
72 }
73 \def\tlet#1#2{\ifx#1\@undefined\let#1=#2\fi}
```

2.5 General markup things

Now for some really simple things. I'll define how to typeset package names and environment names (both in the sans serif font, for now).

```
74 \tlet\package\textsf
75 \tlet\env\textsf
```

I'll define the `\<...>` shortcut for syntax items suggested in the `syntax` package.

```
76 \tdef\<#1>{\synt{#1}}
```

And because it's used in a few places (mainly for typesetting lengths), here's a command for typesetting fractions in text.

```
77 \tdef\smallf#1/#2{\ensuremath{\sim{#1}\!/_{#2}}}
```

2.6 A table environment

`tab` Most of the packages don't use the (obviously perfect) `mdwtab` package, because it's big, and takes a while to load. Here's an environment for typesetting centred tables. The first (optional) argument is some declarations to perform. The mandatory argument is the table preamble (obviously).

```
78 \@ifundefined{tab}{%
79   \newenvironment{tab}[2][\relax]{%
80     \par\vskip2ex%
81     \centering%
82     #1%
83     \begin{tabular}{#2}%
84   }{%
85     \end{tabular}%
86     \par\vskip2ex%
87   }
88 }
```

2.7 Commenting out of stuff

`meta-comment` Using `\iffalse... \fi` isn't much fun. I'll define a gobbling environment using the `sverb` stuff.

```
89 \ignoreenv{meta-comment}
```

2.8 Float handling

This gubbins will try to avoid float pages as much as possible, and (with any luck) encourage floats to be put on the same pages as text.

```
90 \def\textfraction{0.1}
91 \def\topfraction{0.9}
92 \def\bottomfraction{0.9}
93 \def\floatpagefraction{0.7}
```

Now redefine the default float-placement parameters to allow 'here' floats.

```
94 \def\fps@figure{htbp}
95 \def\fps@table{htbp}
```

2.9 Other bits of parameter tweaking

Make grammar environments look pretty, by indenting the left hand sides by a large amount.

```
96 \grammarindent1in
```

I don't like being told by \TeX that my paragraphs are hard to linebreak: I know this already. This lot should shut \TeX up about most problems.

```
97 \sloppy
98 \hbadness\@M
99 \hfuzz10\p@
```

Also make \TeX shut up in the index. The `multicol` package irritatingly plays with `\hbadness`. This is the best hook I could find for playing with this setting.

```
100 \expandafter\def\expandafter\IndexParms\expandafter{%
101   \IndexParms%
102   \hbadness\@M%
103 }
```

The other thing I really don't like is 'Marginpar moved' warnings. This will get rid of them, and lots of other \LaTeX warnings at the same time.

```
104 \let\@latex@warning@no@line\@gobble
```

Put some extra space between table rows, please.

```
105 \def\arraystretch{1.2}
```

Most of the code is at guard level one, so typeset that in upright text.

```
106 \setcounter{StandardModuleDepth}{1}
```

2.10 Contents handling

I use at least one contents file (the main table of contents) although I may want more. I'll keep a list of contents files which I need to handle.

There are two things I need to do to contents files here:

- I must typeset the table of contents at the beginning of the document; and
- I want to typeset tables of contents in two columns (using the `multicol` package).

The list consists of items of the form `\do{<extension>}{<command>}`, where `<extension>` is the file extension of the contents file, and `<command>` is the command to typeset it.

`\docontents` This is where I keep the list of contents files. I'll initialise it to just do the standard contents table.

```
107 \def\docontents{\do{toc}{\tableofcontents}}
```

`\addcontents` By saying `\addcontents{<extension>}{<command>}`, a document can register a new table of contents which gets given the two-column treatment properly. This is really easy to implement.

```
108 \def\addcontents#1#2{%
109   \toks@{\expandafter{\docontents\do{#1}{#2}}}%
110   \edef\docontents{\the\toks@}%
111 }
```

2.11 Finishing it all off

`\finalstuff` The `\finalstuff` macro is a hook for doing things at the end of the document. Currently, it inputs the licence agreement as an appendix.

```
112 \tdef\finalstuff{\appendix\part*{Appendix}\input{gpl}}
```

`\implementation` The `\implementation` macro starts typesetting the implementation of the package(s). If we're not doing the implementation, it just does this lot and ends the input file.

I define a macro with arguments inside the `\StopEventually`, which causes problems, since the code gets put through an extra level of `\def`ing depending on whether the implementation stuff gets typeset or not. I'll store the code I want to do in a separate macro.

```
113 \def\implementation{\StopEventually{\attheend}}
```

Now for the actual activity. First, I'll do the `\finalstuff`. Then, if `doc`'s managed to find the `multicol` package, I'll add the end of the environment to the end of each contents file in the list. Finally, I'll read the index in from its formatted `.ind` file.

```
114 \tdef\attheend{%
115   \finalstuff%
116   \ifhave@multicol%
117     \def\do##1##2{\addtocontents{##1}{\protect\end{multicols}}}%
118     \docontents%
119   \fi%
120   \PrintIndex%
121 }
```


2.12 File version information

`\mdwpkginfo` For setting up the automatic titles, I'll need to be able to work out file versions and things. This macro will, given a file name, extract from L^AT_EX the version information and format it into a sensible string.

First of all, I'll put the original string (direct from the `\Provides...` command). Then I'll pass it to another macro which can parse up the string into its various bits, along with the original filename.

```
122 \def\mdwpkginfo#1{%
123   \edef\@tempa{\csname ver@#1\endcsname}%
124   \expandafter\mdwpkginfo@i\@tempa\@#1\@%
125 }
```

Now for the real business. I'll store the string I build in macros called `\<filename>date`, `\<filename>version` and `\<filename>info`, which store the file's date, version and 'information string' respectively. (Note that the file extension isn't included in the name.)

This is mainly just tedious playing with `\expandafter`. The date format is defined by a separate macro, which can be modified from the configuration file.

```
126 \def\mdwpkginfo@i#1/#2/#3 #4 #5\@#6.#7\@#7{%
127   \expandafter\def\csname #6date\endcsname%
128     {\protect\mdwdateformat{#1}{#2}{#3}}%
129   \expandafter\def\csname #6version\endcsname{#4}%
130   \expandafter\def\csname #6info\endcsname{#5}%
131 }
```

`\mdwdateformat` Given three arguments, a year, a month and a date (all numeric), build a pretty date string. This is fairly simple really.

```
132 \tdef\mdwdateformat#1#2#3{\number#3\ \monthname{#2}\ \number#1}
133 \def\monthname#1{%
134   \ifcase#1\or%
135     January\or February\or March\or April\or May\or June\or%
136     July\or August\or September\or October\or November\or December%
137   \fi%
138 }
139 \def\numsuffix#1{%
140   \ifnum#1=1 st\else%
141   \ifnum#1=2 nd\else%
142   \ifnum#1=3 rd\else%
143   \ifnum#1=21 st\else%
144   \ifnum#1=22 nd\else%
145   \ifnum#1=23 rd\else%
146   \ifnum#1=31 st\else%
147   th%
148   \fi\fi\fi\fi\fi\fi\fi%
149 }
```

`\mdwfileinfo` Saying `\mdwfileinfo{<file-name>}{<info>}` extracts the wanted item of `<info>` from the version information for file `<file-name>`.

```
150 \def\mdwfileinfo#1#2{\mdwfileinfo@i{#2}#1.\@#1}
151 \def\mdwfileinfo@i#1#2.#3\@#1{\csname#2#1\endcsname}
```

2.13 List handling

There are several other lists I need to build. These macros will do the necessary stuff.

`\mdw@ifitem` The macro `\mdw@ifitem<item>\in<list>{\<true-text>}{\<false-text>}` does `<true-text>` if the `<item>` matches any item in the `<list>`; otherwise it does `<false-text>`.

```
152 \def\mdw@ifitem#1\in#2{%
153   \@tempswafalse%
154   \def\@tempa{#1}%
155   \def\do##1{\def\@tempb{##1}\ifx\@tempa\@tempb\@tempswatrue\fi}%
156   #2%
157   \if@tempswa\expandafter\@firstoftwo\else\expandafter\@secondoftwo\fi%
158 }
```

`\mdw@append` Saying `\mdw@append<item>\to<list>` adds the given `<item>` to the end of the given `<list>`.

```
159 \def\mdw@append#1\to#2{%
160   \toks@{\do{#1}}%
161   \toks\tw@\expandafter{#2}%
162   \edef#2{\the\toks\tw@\the\toks@}%
163 }
```

`\mdw@prepend` Saying `\mdw@prepend<item>\to<list>` adds the `<item>` to the beginning of the `<list>`.

```
164 \def\mdw@prepend#1\to#2{%
165   \toks@{\do{#1}}%
166   \toks\tw@\expandafter{#2}%
167   \edef#2{\the\toks@\the\toks\tw@}%
168 }
```

`\mdw@add` Finally, saying `\mdw@add<item>\to<list>` adds the `<item>` to the list only if it isn't there already.

```
169 \def\mdw@add#1\to#2{\mdw@ifitem#1\in#2}{\mdw@append#1\to#2}
```

2.14 Described file handling

I'll maintain lists of packages, document classes, and other files described by the current documentation file.

First of all, I'll declare the various list macros.

```
170 \def\dopackages{}
171 \def\doclasses{}
172 \def\dotherfiles{}
```

`\describepackage` A document file can declare that it describes a package by saying `\describepackage{<package-name>}`. I add the package to my list, read the package into memory (so that the documentation can offer demonstrations of it) and read the version information.

```
173 \def\describepackage#1{%
174   \mdw@ifitem#1\in\dopackages}{%
175     \mdw@append#1\to\dopackages%
176     \usepackage{#1}%
```

```

177     \mdwpkginfo{#1.sty}%
178   }%
179 }

\describessclass By saying \describessclass{<class-name>}, a document file can declare that it
describes a document class. I'll assume that the document class is already loaded,
because it's much too late to load it now.

180 \def\describessclass#1{\mdw@add#1\to\doclasses\mdwpkginfo{#1.cls}}

\describessfile Finally, other 'random' files, which don't have the status of real LATEX packages
or document classes, can be described by saying \describessfile{<file-name>} or
\describessfile*{<file-name>}. The difference is that the starred version will not
\input the file.

181 \def\describessfile{%
182   \@ifstar{\describessfile@i\@gobble}{\describessfile@i\input}%
183 }
184 \def\describessfile@i#1#2{%
185   \mdw@ifitem#2\in\dootherfiles{ }{%
186     \mdw@add#2\to\dootherfiles%
187     #1{#2}%
188     \mdwpkginfo{#2}%
189   }%
190 }

```

2.15 Author and title handling

I'll redefine the `\author` and `\title` commands so that I get told whether I need to do it myself.

```

\author This is easy: I'll save the old meaning, and then redefine \author to do the old
thing and redefine itself to then do nothing.

191 \let\mdw@author\author
192 \def\author{\let\author\@gobble\mdw@author}

\title And oddly enough, I'll do exactly the same thing for the title, except that I'll also
disable the \mdw@builddtitle command, which constructs the title automatically.

193 \let\mdw@title\title
194 \def\title{\let\title\@gobble\let\mdw@builddtitle\relax\mdw@title}

\date This works in a very similar sort of way.

195 \def\date#1{\let\date\@gobble\def\today{#1}}

\datefrom Saying \datefrom{<file-name>} sets the document date from the given filename.

196 \def\datefrom#1{%
197   \protected@edef\@tempa{\noexpand\date{\csname #1date\endcsname}}%
198   \@tempa%
199 }

\docfile Saying \docfile{<file-name>} sets up the file name from which documentation
will be read.

200 \def\docfile#1{%

```

```

201 \def\@tempa##1.##2\@{\def\@basefile{##1.##2}\def\@basename{##1}}%
202 \edef\@tempb{\noexpand\@tempa#1\noexpand\@}%
203 \@tempb%
204 }

```

I'll set up a default value as well.

```

205 \docfile{\jobname.dtx}

```

2.16 Building title strings

This is rather tricky. For each list, I need to build a legible looking string.

`\mdw@addtotitle` By saying `\mdw@addtotitle{<list>}{<command>}{<singular>}{<plural>}` I can add the contents of a list to the current title string in the `\mdw@title` macro.

```

206 \tdef\mdw@addtotitle#1#2#3#4{%

```

Now to get to work. I need to keep one 'lookahead' list item, and a count of the number of items read so far. I'll keep the lookahead item in `\@nextitem` and the counter in `\count@`.

```

207 \count@\z0%

```

Now I'll define what to do for each list item. The `\protect` command is already set up appropriately for playing with `\edef` commands.

```

208 \def\do##1{%

```

The first job is to add the previous item to the title string. If this is the first item, though, I'll just add the appropriate 'The ' or ' and the ' string to the title (this is stored in the `\@prefix` macro).

```

209 \edef\mdw@title{%
210 \mdw@title%
211 \ifcase\count@\@prefix%
212 \or\@nextitem%
213 \else, \@nextitem%
214 \fi%
215 }%

```

That was rather easy. Now I'll set up the `\@nextitem` macro for the next time around the loop.

```

216 \edef\@nextitem{%
217 \protect#2{##1}%
218 \protect\footnote{%
219 The \protect#2{##1} #3 is currently at version %
220 \mdwfileinfo{##1}{version}, dated \mdwfileinfo{##1}{date}.%
221 }\space%
222 }%

```

Finally, I need to increment the counter.

```

223 \advance\count@\@ne%
224 }%

```

Now execute the list.

```

225 #1%

```

I still have one item left over, unless the list was empty. I'll add that now.

```

226 \edef\mdw@title{%
227   \mdw@title%
228   \ifcase\count@%
229   \or\@nextitem\space#3%
230   \or\ and \@nextitem\space#4%
231   \else,\ and \@nextitem\space#4%
232   \fi%
233 }%
```

Finally, if $\text{\count@} \neq 0$, I must set \@prefix to ‘ and the ’.

```

234 \ifnum\count@>\z@\def\@prefix{ and the }\fi%
235 }
```

\mdw@buildtitle This macro will actually do the job of building the title string.

```

236 \tdef\mdw@buildtitle{%
```

First of all, I'll open a group to avoid polluting the namespace with my gubbins (although the code is now much tidier than it has been in earlier releases).

```

237 \begingroup%
```

The title building stuff makes extensive use of \edef . I'll set \protect appropriately. (For those not in the know, $\text{\@unexpandable@protect}$ expands to ‘ $\text{\noexpand\protect\noexpand}$ ’, which prevents expansion of the following macro, and inserts a \protect in front of it ready for the next \edef .)

```

238 \let\@@protect\protect\let\protect\@unexpandable@protect%
```

Set up some simple macros ready for the main code.

```

239 \def\mdw@title{}%
240 \def\@prefix{The }%
```

Now build the title. This is fun.

```

241 \mdw@addtotitle\dopackages\package{package}{packages}%
242 \mdw@addtotitle\doclasses\package{document class}{document classes}%
243 \mdw@addtotitle\dotherfiles\texttt{file}{files}%
```

Now I want to end the group and set the title from my string. The following hacking will do this.

```

244 \edef\next{\endgroup\noexpand\ttitle{\mdw@title}}%
245 \next%
246 }
```

2.17 Starting the main document

\mdwdoc Once the document preamble has done all of its stuff, it calls the \mdwdoc command, which takes over and really starts the documentation going.

```

247 \def\mdwdoc{%
```

First, I'll construct the title string.

```

248 \mdw@buildtitle%
249 \author{Mark Wooding}%
```

Set up the date string based on the date of the package which shares the same name as the current file.

```
250 \datefrom\@basename%
```

Set up verbatim characters after all the packages have started.

```
251 \shortverb\|%
```

```
252 \shortverb\"%
```

Start the document, and put the title in.

```
253 \begin{document}
```

```
254 \maketitle%
```

This is nasty. It makes maths displays work properly in demo environments. *The L^AT_EX Companion* exhibits the bug which this hack fixes. So ner.

```
255 \abovedisplayskip\z@%
```

Now start the contents tables. After starting each one, I'll make it be multi-columnar.

```
256 \def\do##1##2{%
```

```
257   ##2%
```

```
258   \ifhave@multicol\addtocontents{##1}{%
```

```
259     \protect\begin{multicols}{2}%
```

```
260     \hbadness\@M%
```

```
261   }\fi%
```

```
262 }%
```

```
263 \docontents%
```

Input the main file now.

```
264 \DocInput{\@basefile}%
```

That's it. I'm done.

```
265 \end{document}
```

```
266 }
```

2.18 And finally...

Right at the end I'll put a hook for the configuration file.

```
267 \ifx\mdwhook\@undefined\else\expandafter\mdwhook\fi
```

That's all the code done now. I'll change back to 'user' mode, where all the magic control sequences aren't allowed any more.

```
268 \makeatother
```

```
269 \</mdwtools>
```

Oh, wait! What if I want to typeset this documentation? Aha. I'll cope with that by comparing `\jobname` with my filename `mdwtools`. However, there's some fun here, because `\jobname` contains category-12 letters, while my letters are category-11. Time to play with `\string` in a messy way.

```
270 \<driver>
```

```
271 \makeatletter
```

```
272 \edef\@tempa{\expandafter\@gobble\string\mdwtools}
```

```
273 \edef\@tempb{\jobname}
```

```
274 \ifx\@tempa\@tempb
```

```

275 \describesfile*{mdwtools.tex}
276 \docfile{mdwtools.tex}
277 \makeatother
278 \expandafter\mdwdoc
279 \fi
280 \makeatother
281 \</driver>

```

That's it. Done!

Mark Wooding, 10 May 1996

Appendix

A The GNU General Public License

The following is the text of the GNU General Public License, under the terms of which this software is distributed.

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright © 1989, 1991 Free Software Foundation, Inc.

51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make

sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

- 3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
 - (a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
 - (b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

- (c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.
6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.
7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by

all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT

HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

one line to give the program’s name and a brief idea of what it does.
Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) yyyy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for de-
tails type 'show w'.
This is free software, and you are welcome to redistribute it under
certain conditions; type 'show c' for details.
```

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
'Gnomovision' (which makes passes at compilers) written by James
Hacker.
```

```
signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Symbols		
<code>\!</code>	77	<code>\@ifpackageloaded</code> 8–10
<code>\"</code>	252	<code>\@ifstar</code> 14, 182
<code>\<</code>	2, 76	<code>\@ifundefined</code> 78
<code>\@@</code>	124, 126, 150, 151, 201, 202	<code>\@input</code> 3
<code>\@@protect</code>	238	<code>\@latex@warning@no@line</code> 104
<code>\@@undefined</code>	19, 67, 73, 267	<code>\@nextitem</code> 212, 213, 216, 229–231
<code>\@M</code>	98, 102, 260	<code>\@prefix</code> 211, 234, 240
<code>\@basefile</code>	201, 264	<code>\@secondoftwo</code> 157
<code>\@basename</code>	201, 250	<code>\@tempa</code> 123, 124, 154,
<code>\@firstoftwo</code>	157	155, 197, 198, 201, 202, 272, 274
<code>\@gobble</code>	65, 104, 182, 192, 194, 195, 272	<code>\@tempb</code> 155, 202, 203, 273, 274
<code>\@gobbledef</code>	<u>65</u> , 70	<code>\@tempswafalse</code> 24, 153
		<code>\@tempswatrue</code> 25–28, 155

\@twoclasseserror	4	environments:	
\@unexpandable@protect	238	meta-comment	89
\	251	tab	78
F			
_	13, 132, 230, 231	\finalstuff	112, 115
A			
\abovedisplayskip	255	\floatpagefraction	93
\active	13	\footnote	218
\addcontents	2, 108	\fps@figure	94
\addtocontents	117, 258	\fps@table	95
G			
\appendix	112	\grammarindent	96
\arraystretch	105	H	
\attheend	113, 114	\hbadness	98, 102, 260
\author	2, 191, 249	\hfuzz	99
B			
\begin	83, 253, 259	I	
\bottomfraction	92	\if@tempswa	29, 157
C			
\centering	81	\ifcreateindex	18, 34, 41
\CodelineIndex	42	\ifhave@multicol	116, 258
\CodelineNumbered	45	\ignoreenv	89
\count@	207, 211, 223, 228, 234	\implementation	2, 113
\createindexfalse	27, 28	\in	152, 169, 174, 185
\createindextrue	25, 26	\indexing	3, 19, 21, 25–28, 31
D			
\date	2, 195, 197	\IndexParms	100, 101
\datefrom	196, 250	\input	112, 182
\describesclass	2, 180	L	
\describesfile	2, 181, 275	\loop	23
\describesfile*	2	M	
\describesfile@i	182, 184	\makeatletter	2, 271
\describespackage	2, 173	\makeatother	268, 277, 280
\DisableCrossrefs	46	\maketitle	254
\do	107, 109, 117, 155, 160, 165, 208, 256	\mdw@add	169, 180, 186
\docdate	2	\mdw@addtotitle	206, 241–243
\docfile	200, 276	\mdw@append	159, 169, 175
\DocInput	264	\mdw@author	191, 192
\doclasses	171, 180, 242	\mdw@buildtitle	194, 236, 248
\docontents	107, 109, 110, 118, 263	\mdw@ifitem	152, 169, 174, 185
\documentclass	4, 5	\mdw@prepend	164
\doneclasses	6	\mdw@title	193, 194, 209, 210, 226, 227, 239, 244
\DoNotIndex	48, 49, 51, 53–57, 59–63	\mdw@undefined	6
\doootherfiles	172, 185, 186, 243	\mdwdateformat	3, 128, 132
\dopackages	170, 174, 175, 241	\mdwdoc	247, 278
E			
\EnableCrossrefs	43	\mdwfileinfo	150, 220
\end	85, 117, 265	\mdwfileinfo@i	150, 151
\ensuremath	77	\mdwhook	3, 267
\env	75	\mdwpkginfo	122, 177, 180, 188
		\mdwpkginfo@i	124, 126
		\mdwtools	272
		\mdwtype	12, 20, 30, 33, 35, 36, 38, 40

<code>\mdwtype@i</code>	14, 17	<code>\shortverb</code>	251, 252
<code>meta-comment (environment)</code>	<u>89</u>	<code>\sixt@cn</code>	14, 31
<code>\monthname</code>	3, 132, 133	<code>\sloppy</code>	97
N		<code>\smallf</code>	2, 77
<code>\newenvironment</code>	79	<code>\StopEventually</code>	113
<code>\newif</code>	18	<code>\synt</code>	76
<code>\next</code>	244, 245	T	
<code>\number</code>	132	<code>tab (environment)</code>	<u>78</u>
<code>\numsuffix</code>	3, 139	<code>\tableofcontents</code>	107
O		<code>\tdef</code> <u>66</u> , 76, 77, 112, 114, 132, 206, 236	
<code>\obeyspaces</code>	11	<code>\textfraction</code>	90
P		<code>\textsf</code>	74, 75
<code>\package</code>	74, 241, 242	<code>\texttt</code>	243
<code>\par</code>	80, 86	<code>\title</code>	2, <u>193</u> , 244
<code>\part</code>	112	<code>\tlet</code>	<u>66</u> , 74, 75
<code>\PrintIndex</code>	120	<code>\to</code>	159, 164, 169, 175, 180, 186
<code>\protect</code> ..	117, 128, 217–219, 238, 259	<code>\today</code>	195
<code>\protected@edef</code>	197	<code>\toks</code>	161, 162, 166, 167
R		<code>\toks@</code>	109, 110, 160, 162, 165, 167
<code>\repeat</code>	32	<code>\topfraction</code>	91
S		U	
<code>\setcounter</code>	64, 106	<code>\usepackage</code>	8–10, 176