

A Few Packages by Donald Arseneau

Uwe Lück*

March 18, 2012

Abstract

This document demonstrates the present (i.e., `nicetext` bundle release v0.4) capabilities of `makedoc` and `niceverb` to typeset L^AT_EX quality documentation from third-party package files having ASCII comments only, without modifying such package files manually.¹ Such packages usually mark comments by lines starting with ‘%_L’. This is somewhat difficult when the “comment mark” ‘%’ may at the same time be used for “commenting out” in place of “true commenting.” As opposed to `substr.sty` that conforms to the easier ‘%_L’ style, the packages `nolbreaks` and `notoccite` by Donald Arseneau exemplify the ‘%_L’ commenting style.

The ASCII documentation of packages of this kind does not always clearly indicate when switching to `\tt` is appropriate. This problem is solved here by string replacements very specific to the package files. The setup for these replacements (in the source file `arseneau.tex`) still may be simplified.

Moreover, some similar packages cannot be properly typeset with `nicetext` at present because `makedoc`’s loop is too rigid to deal with (i) ASCII indents, displays, lists and with (ii) instructions after `\endinput`. (On 2010/04/05, we try addressing (i) with `wiki.sty`, applied to Donald Arseneau’s `optional`. A `nicetext` bug with replacing tildes shows up.)

It should be clear that all the package descriptions are Donald Arseneau’s, `nicetext` just formats them.

Contents

1	nolbreaks	2
2	notoccite	3
3	optional	4

*applying <http://ctan.org/pkg/nicotext>

¹See <http://ctan.org/pkg/nicotext> for more on these packages.

1 nolbreaks

```

1 % nolbreaks.sty by Donald Arseneau
2 % Public domain software -- please improve and submit to CTAN
3
4 \ProvidesPackage{nolbreaks}[2002/09/19 \space v 1.0 \space
5   - no linebreaks in text]
6

```

Use `\nolbreaks{some text}` to prevent linebreaks in `some␣text`. This has the advantage over `\mbox{}` that glue (rubber space) remains flexible. It has the disadvantage of not working in all cases! Most common cases are handled here (`\linebreak` is disabled, for example) but spaces hidden in macros or `{␣}` can still create break-points.

Large pieces of text with no breaks can cause problems with paragraph justification. Giving the package option `[ragged]` allows a line before the unbreakable text to be cut short.

You should declare `\sloppy` in your document.

```

7 \let\nb@ragged\relax
8
9 \DeclareOption{ragged}{
10   \def\nb@ragged{%
11     \skip@\lastskip \unskip
12     \nb@counter \lastpenalty
13     \hskip \z@ plus 2cm\relax
14     \penalty\nb@counter
15     \advance\skip@ \z@ plus -2cm\relax
16     \hskip\skip@
17   }}
18 \ProcessOptions
19
20 \let\nb@@iwspace=\ %
21 \let\nb@@hskip=\hskip
22 \let\nb@@penalty=\penalty
23 \newcount\nb@counter
24 \ifx\langwohyphens\undefined
25   \newlanguage\nb@lang
26 \else
27   \let\nb@lang\langwohyphens
28 \fi
29
30 \DeclareRobustCommand{\nolbreaks}[1]{%
31   \leavevmode
32   \begingroup
33     % Apply flexible lead-in
34     \nb@ragged
35     % Prevent hyphenation

```

```

36     \language\nb@lang
37     % Disable commands that give breakpoints
38     \let\ \nb@iwspace
39     \let\hskip\nb@hskip
40     \let\penalty\nb@penalty
41     \let\language\nb@counter
42     % Prevent breaks in math
43     \relpenalty\M
44     \binoppenalty\M
45     % Prevent breaks at spaces (only outermost visible spaces)
46     \@PreserveSpaces \@empty #1 \@PreserveSpaces
47 \endgroup
48 }
49
50 \def\@PreserveSpaces#1 {#1\@ifnextchar\@PreserveSpaces{\@gobble}%
51   {\@firstofone{\nb@@penalty\M} \@PreserveSpaces\@empty}}
52
53 \def\nb@iwspace{\nb@@penalty\M \nb@@iwspace}
54 \def\nb@hskip {\nb@@penalty\M \nb@@hskip}
55 \def\nb@penalty{\nb@@penalty\M \nb@counter}
56

```

2 notocite

```

1 % notocite.sty      no t.o.c. cite      Jul 20, 2000
2 % Donald Arseneau  asnd@triumf.ca      TRIUMF, Vancouver, Canada,
3 % This is unrestricted software contributed to the public domain.

```

Ordinarily, cites used in titles or figure captions also appear in the table of contents and list of figures. If you then run `bibtex` using the `unsrt` (unsorted) style, they get numbered starting from 1, not the number they should have in the main text.

A good option is to avoid cites in titles, and to specify optional caption text without cites:

```
\caption[Picture of a bird.]{Picture of a bird \cite{audobon}.}
```

If you must use moving cites, you could manage them by deleting `.toc` and `.lof` files, then running `latex` once, then `bibtex`. However, the following definition fixes the problem so you don't need to worry about that.

NOTE: This definition works for the ordinary \LaTeX definitions for `\cite` and others (`\addtocontents`, `\label`) but it may well fail when used with various packages for citations or cross references.

It works by locally setting `\@fileswfalse`, which is something like `\nofiles`, but `\@fileswfalse` does not affect `\label` or `\addtocontents`. `\nofiles` does

most of its work by redefining `\protected@write`, and neither `\addtocontents` nor `\label` check for `\if@filesw`. `\cite` *does* check `\if@filesw`.

```

4 \ProvidesPackage{notoccite}[2000/07/20]
5 \def\@starttoc#1{%
6   \begingroup
7     \@fileswfalse
8     \makeatletter
9     \@input{\jobname.#1}%
10  \endgroup
11  \if@filesw
12    \expandafter\newwrite\csname tf@#1\endcsname
13    \immediate\openout \csname tf@#1\endcsname \jobname.#1\relax
14  \fi
15  \@nobreakfalse
16 }
17
18
```

3 optional

```

1 %
2 % O P T I O N A L . S T Y
3 % ~~~~~
4 % ver 2.2b Jan 2005
5 %
6 % Enable multiple versions of a document to be printed from one source file,
7 % especially if most of the text is shared between versions.
8 %
9 % Copyright 1993,1999,2001,2005 by Donald Arseneau (asnd@triumf.ca).
10 % This software is released under the terms of the LaTeX Project Public
11 % License (ftp://ctan.tug.org/tex-archive/macros/latex/base/lppl.txt).
12 % (Essentially: Free to use, copy, distribute (sell) and change, but, if
13 % changed, that fact must be made apparent to the user.) It has a
14 % status of maintained.
```

How to Use

One way to use this package is to declare (for example)

```
\usepackage[opta]{optional}
```

at the beginning of your document, and flag optional text throughout your document like:

```

\opt{opta}{Do this if option opta was declared}
\opt{optb}{Do this if option optb was declared}
\opt{optx,opty}{Do this if either option optx or opty}
\opt{}{Never print this text!}
```

```
\opt{opta}{\input{appendices}}
\optv{xam}{Type: \verb|[root /]$ rm -r *|.}
```

Note that both the package option and the `\opt` argument can contain lists of options although, in practice, one or the other should be a single option name. Lists are allowed in both places to allow more flexibility in the style of use. (But making the definitions much more difficult, Grrr.)

Just as for `\includeonly`, you will have to edit the main document file to switch option codes (i.e., change the `\usepackage` line). There are, however, several ways to use this package without altering the main document file: separate files, file-name sensing, interactive prompting, and command-line option selection.

Typically, different versions of a document will require different document class and package setup, besides the different tags for `optional.sty`. In that case it is best to have a separate main file for each version of the document. Each stub file will declare the document class and load some packages (including this one) and then input the rest of the document from a file common to all versions.

```
\documentclass[A0]{poster}
\usepackage[poster]{optional}
\input{my_paper}
```

If the different `opt`-tags match the different stub file names (file `poster.tex` will typeset the `poster` version) then you can specify

```
\usepackage[\jobname]{optional}
```

Alternatively, this `\jobname` technique can make use of symbolic links, if your computer system supports them, by having a single main input file accessed under different names (and different `\jobnames`).

Another scheme is to invoke \LaTeX with the command line such as:

```
latex \def\UseOption{opta,optb}\input{file}
```

(with quoting appropriate to your operating system) then options `opta` and `optb` will be used in addition to any options specified with the `\usepackage` command.

You can prompt yourself to specify the option(s) with every run through \LaTeX :

```
\usepackage{optional}
\newcommand{\ExplainOptions}{man = users manual, check = checklist,
    ref = reference card, post = poster.}
\AskOption
```

The definition of `\ExplainOptions` is optional; it only serves to help the person who answers the question. The `\AskOption` is also optional; it will be executed automatically whenever `optional.sty` sees no list of options. This method is too tedious to use much.

The normal restrictions forbidding special characters in package options and reference tags apply also to the tags used by the `\opt` command.

These are not `comment` macros: The optional text must be well-formed with balanced braces, even if not printed. The `\opt` command *is* completely **expandable** which means it is robust and can even be used in messages (`\typeout`).

As usual, `\verb` commands and verbatim environments cannot be used in the argument to `\opt`. For this purpose there is a variant form of `\opt` called `\optv` (optional verbatim) which may have a limited class of verbatim material in the argument. It can do so by leaving the braces around the argument, which may have undesired side effects. For an `\optv` argument to be successfully ignored, the verbatim material must have balanced braces etc.

The `\opt` command is only intended for small sections of text. If you need to optionally include whole sections or chapters, put that material in a separate file, and `\opt`-ionally use an `\input` command:

```

\opt{internal}{\input{prog_listings}}
15 %===== END INSTRUCTIONS =====
16
17 \ProvidesPackage{optional}[2005/01/26 ver 2.2b; \space
18   Optional inclusion/omission]

```

Initialize used-option-list to `\@gobble` to eat the comma when the first entry is appended.

```

19 \ifundefined{UseOption}{\let\UseOption\@gobble}{}
20 \DeclareOption*{\edef\UseOption{\UseOption,\CurrentOption}}
21 \ProcessOptions
22 \AtBeginDocument{\Opl@Setup}
23
24 \newcommand*\opt[1]{\if\Opl@notlisted{#1}\expandafter\@gobble
25   \else \expandafter\@firstofone \fi}
26
27 \newcommand*\optv[1]{\if\Opl@notlisted{#1}\expandafter\@gobble\fi}
28

```

This initial definition forces immediate setup if `\opt` used in the preamble

```

29 \def\Opl@notlisted{\fi \Opl@Setup \if\Opl@notlisted}
30
31 \newcommand\AskOption{%
32   \@ifundefined{ExplainOptions}{}{\typeout{\ExplainOptions}}%
33   \typein[\UseOption]{Specify which optional text to process:}%
34   }
35
36 \def\Opl@Setup{%
37   \ifx\UseOption\@gobble\AskOption\fi
38   \let\Opl@notlisted\empty % initialize list of checks
39   \@for\@tempa:=\UseOption\do{%

```

```

40     \ifx\@tempa\@empty\else\expandafter\Opl@oneop\expandafter{\@tempa}\fi}%
41 \ifx\Opl@notlisted\@empty \PackageWarning{optional}%
42   {No options were selected, so all optional text will be printed}%
43   \let\opt\@secondoftwo
44 \else
45   \typeout{Using optional text marked with \UseOption. }%
46   \toks@\expandafter{\Opl@notlisted}%
47   \edef\@tempa{\def\noexpand\Opl@notlisted####1{\the\toks@,}}\@tempa
48 \fi
49 \let\Opl@Setup\@empty \let\Opl@oneop\undefined
50 \let\AskOption\undefined \let\ExplainOptions\undefined
51 }
52 \begingroup
53 \catcode'\Z= 3 % special delimiter
54 \gdef\Opl@oneop#1{%
55   \@ifundefined{Opl@Match@#1}{%
56     \toks@\expandafter{\Opl@notlisted}%
57     \edef\Opl@notlisted{\the\toks@ \csname Opl@Match@#1\endcsname ,####1,#1,Z}%
58     \@namedef{Opl@Match@#1}##1,#1,##2Z{##2}%
59   }\relax
60 }
61 \endgroup
62 \endinput
63

```