The nag package[*]

Dr. Ulrich Michael Schwarz[†]

November 27, 2011

Abstract

Old habits die hard. All the same, there are commands, classes and packages which are outdated and superseded. nag provides routines to warn the user about the use of those. As an example, we provide an extension that detects many of the "sins" described in l2tabu.

Contents

---

[*] This document corresponds to nag 0.7, dated 2011/11/19. Other versions can be found at http://absatzen.de/

[†] `ulmi@absatzen.de`

1 User-side considerations.

1.1 Installation.

Process `nag.ins` with LaTeX to obtain some files: `nag.sty` and `nag-l2tabu.cfg` et al. must go to a place where LaTeX will find them, like the local TEXMF tree. (If all else fails and you need it to work *right now*, having them in the same directory as the LaTeX file you want to use them on may work under many circumstances.) You can, as usual, run LaTeX on `nag.dtx` to obtain this documentation, including the implemenation docs. (This is recommended if you plan to extend `nag` to handle your own packages.) `nagdemo.tex` is a horrible document that will show you many of the warnings that `nag` can generate.

1.2 Usage.

Add the following to the beginning your main document (Comments and `\listfiles` can be safely left before it, though):

```
\RequirePackage[l2tabu, orthodox]{nag}
```

This will check for many common mistakes, and give some hints on what to use instead. However, you should always refer to l2tabu for a more detailed explanation of the whats and whys: it gives more information than can be possibly pressed into two lines of error message. Orthodox checks for pitfalls that are not technically incorrect. If you know what you're doing, omit orthodox.

1.3 Known bugs

currently none.

1.4 `nag-l2tabu.cfg`

In a nutshell, `nag-l2tabu.cfg` detects the following:

- Usage of the 2.09-style font commands `\it`, `\bf`, `\rm`, `\sc`, `\sl`, `\tt` and `\cal`.
- Usage of `\centerline`.
- Usage of the outdated packages epsfig, psfig, epsf, doublespace, fancyheadings, scrpage, umlaut, isolatin, isolatin1, t1enc, caption2, psfonts, mathptm, times, palatino, mathpple, euler and utopia, and of the outdated class scrlttr.
- Figures and tables without caption (this is not technically in l2tabu, but the people who have floats without captions tend to ask "Why is LaTeX moving my pictures away from where I put them?"), labels within floats that do not reference the caption, and usage of the center environment within floats.

It is beyond the possibilities of this package to detect things like use of TeX assignment syntax, or direct change of paper parameters, or reliable detection of user-issued `\sloppy`. eqnarray is handled as of 0.60alpha4, and there is code for $$ in experimental since 0.60alpha4, which has been moved to l2tabu in 0.60.

Be warned, that this package will possibly balk at legitimate use, and not find illegitimate use in all cases. It is a tool, not a replacement for study of l2tabu.

```
\ProvidesFile{nag-l2tabu.cfg}
             [2010/05/17 v2.11 l2tabu rules for nag.sty (ulmi)]
%%
%% The sins.
%%
%% Section numbers refer to l2tabuen 1.7 revised/enlarged dated 2004OCT24
%% \S 1.1
\ObsoletePackage{a4wide}{the \lq a4paper\rq\space class option}
\ObsoletePackage{a4}{the \lq a4paper\rq\space class option}
%% \S 1.2--1.5 cannot reasonably be checked programmatically
%% \S 1.6
```

Hacking galore ahead! We will make the dollar active. Since unlike onlyamsmath, we do not change the user's command to LaTeX or amsmath commands, we need to store the old double dollar sequence as well as the single dollar.

```
\def\nag@doubledollar{$$}%$$
\def\nag@singledollar{$}%$
```

This is used to hide our redefinition in unprotected expanding context. This should not happen: you are expected to *always* use protected means of expansion in LaTeX, but fecal matter happens. See below for a good trick to distinguish expansion from executing context.

```
\def\nag@expanding@voodoo#1#2#3{\relax\relax\nag@singledollar}

\def\nag@maybedispmath{%
  \texorpdfstring{%
    %% in TeX context, do tricky stuff.
    \ifinner\expandafter\@firstoftwo
    \else\expandafter\@secondoftwo\fi
    {%% in inner mode, $$ is an empty formula, so no testing wanted.
      \nag@singledollar}%
    {%%
      \ifx\protect\@typeset@protect\expandafter\@firstoftwo
      \else\expandafter\@secondoftwo\fi
      {%% normal case: looks like typesetting
        %% protect against strictly expanding context
        %% like TeX' \message: the first expanding voodoo will expand,
        %% removing the rest, inserting \relax\relax$ instead. This is
        %% not totally transparent, but \let\relax\relax is as close
        %% to a no-op as we can get.
        \let\nag@expanding@voodoo\nag@expanding@voodoo
        \protect\nag@maybe@dispmath}%
      {%% some other case, hide ourselves
        \nag@singledollar}%
    }%
  }{%
    %% in pdf context, just be a math shift. This creates the "math
    %% shift not allowed" warnings we all love.
    \nag@singledollar
  }%
}
```

If the user doesn't load `hyperref`, we have to fake its `\texorpdfstring` command. Note that this will break any package that is foolish enough to detect `hyperref` by testing for definedness of `\texorpdfstring`.

```
\AtBeginDocument{\providecommand\texorpdfstring{\@firstoftwo}}
\AtBeginDocument{\catcode‘$\active}%$
\AtEndDocument{\catcode‘$=3\relax}
```

Now, the proper testing. (Yes, the above is just the technicalities.) We use the kernel's `\@ifnextchar` to look for a possible second dollar. Note however, this would allow skipping of spaces between them, and $_$ is not a displayed equation start in TeX. We work around this by re`\let`ting `\@sptoken` to something that cannot legally appear in the source.

```
\def\nag@quark{\nag@quark}
\bgroup
  \catcode‘$\active%$
  \gdef\nag@maybe@dispmath{%
    \bgroup
    \let\@sptoken\nag@quark% prevent skipping of spaces
    \@ifnextchar${%$%
      \ifmmode
      % we already warned upon entering.
      \else
      \nag@warn{%
        \nag@doubledollar...\nag@doubledollar\space is obsolete.\MessageBreak
        Use \string\[...\string\] et al. instead}%
      \fi
      \egroup\expandafter\nag@doubledollar\@gobble
    }{%
     \egroup\nag@singledollar
    }%
  }
  % we do the assignment here, which means any package that redefines
  % \$ as well will silently disable us. This is a feature.
  \global\let$\nag@maybedispmath%$
\egroup
```

new in 2.1alpha1: more compat testing. Version control keywords are dollar-delimited. all five implementations get it wrong.

```
\AtBeginDocument{%
  \@ifpackageloaded{rcs}{%
```

```
    % this redefinition is functionally equivalent,
    % but does not share actual code.
    \renewcommand\RCS{\bgroup%
      \catcode‘\_ =\active
      \catcode‘\$=3 % this line added for compatibility.
      \csname RCS_get_argument\endcsname
    }
    \PackageInfo{nag}{rcs.sty hack applied}%
  }{}%
  \@ifpackageloaded{svninfo}{%
    \g@addto@macro\@svnBeginRead{\catcode‘\$ 3 }%
    \PackageInfo{nag}{svninfo.sty hack applied}%
  }{}%
  \@ifpackageloaded{svn}{%
    \PackageInfo{nag}{svn.sty is broken: disabling dollar check}%
    \catcode‘\$ 3
  }{}%
  \@ifpackageloaded{rcsinfo}{%
    \PackageInfo{nag}{rcsinfo.sty is broken: disabling dollar check}%
    \catcode‘\$ 3
  }{}%
  \@ifpackageloaded{pgf}{%
    \PackageInfo{nag}{pgf.sty is broken: disabling dollar check}%
    \catcode‘\$ 3
  }{}%
}


%% \S 1.7 cannot reasonably be checked programmatically
%% \S 1.8 \sloppy is called by parbox, among others, and would
%% give many spurious warnings.
%% \S 2.1.1
\ObsoleteCS[an old LaTeX 2.09 command]{bf}
           {\protect\bfseries\space or \protect\textbf}
\ObsoleteCS[an old LaTeX 2.09 command]{it}
           {\protect\itshape\space or \protect\textit}
\ObsoleteCS[an old LaTeX 2.09 command]{rm}
           {\protect\rmfamily\space or \protect\textrm}
\ObsoleteCS[an old LaTeX 2.09 command]{sc}
           {\protect\scshape\space or \protect\textsc}
\ObsoleteCS[an old LaTeX 2.09 command]{sf}
           {\protect\sffamily\space or \protect\textsf}
\ObsoleteCS[an old LaTeX 2.09 command]{sl}
           {\protect\slshape\space or \protect\textsl}
\ObsoleteCS[an old LaTeX 2.09 command]{tt}
           {\protect\ttfamily\space or \protect\texttt}
\ObsoleteCS[an old LaTeX 2.09 command]{cal}
           {\protect\mathcal}% Hmm, this is not in l2tabu?
%% \S 2.1.2
%% Gone with 1.8 because this never worked for the kernel \frac anyway.
%% \ObsoleteCS[TeX]{over}{\protect\frac}
%% \ObsoleteCS[TeX]{choose}{\protect\frac\space or amsmath’s \protect\binom}
%% \S 2.1.3
\ObsoleteCS[TeX]{centerline}{\protect\centering\space or center environment}
%% \S 2.2.1
\ObsoleteClass{scrlettr}{the scrlttr2 package}
%% \S 2.2.2
\ObsoletePackage{epsf}{the graphicx package}
\ObsoletePackage{psfig}{the graphicx package}
\ObsoletePackage[deprecated]{epsfig}{the graphicx package directly}
%% \S 2.2.3
\ObsoletePackage{doublespace}{the setspace package}
%% \S 2.2.4
\ObsoletePackage{fancyheadings}{the fancyhdr or scrpage2 packages}
\ObsoletePackage{scrpage}{the scrpage2 package}
%% \S 2.2.5
\ObsoletePackage{isolatin}{the inputenc package with option latin1}
\ObsoletePackage{umlaut}{the inputenc package with suitable option
  (latin1, utf8 ...)}
\ObsoletePackage{isolatin1}{the inputenc package with option latin1}
%% \S 2.2.6
\ObsoletePackage{t1enc}{the fontenc package with option T1}
%% \S 2.2.7 we don’t check for bst yet.
%% (This is in l2tabu 1.8)
\ObsoletePackage{caption2}{the caption package v3.0 or later}
%% \S 2.3.1-3
\ObsoletePackage{times}
                {the mathptmx, helvet (option scaled=.9), courier packages}
\ObsoletePackage{pslatex}
                {the mathptmx, helvet (option scaled=.9), courier packages}
\ObsoletePackage{mathptm}
                {the mathptmx package}
%% \S 2.3.4-5
\ObsoletePackage{palatino}
                {the mathpazo, helvet (option scaled=.95), courier packages}
\ObsoletePackage{mathpple}{the mathpazo package}
%% \S 2.3.6 can’t be checked
%% \S 2.3.7
\ObsoletePackage{euler}{the eulervm package}
\ObsoletePackage{utopia}{the fourier package}
%% \S 3.1
\NagDeclareFloat{figure}\NagDeclareFloat{table}%
\g@addto@macro\nag@labels{,label,caption@xlabel}%
% \changes{0.60}{2007/03/31}{alternate center-in-float check, doesn’t
% take up as many macro names}
\nag@prepend{endcenter}{%
  \ifx\@captype\@undefined\else
    \nag@warn{\lq center\rq\space environment in \@captype.\MessageBreak
      Maybe you want \protect\centering\space instead}%
  \fi
}%
%% The latter two are used by KOMA-Script, the last by hypcap.
% \changes{0.53}{2007/03/21}{hypcap support. (H.G.Krauth\"auser)}
% \changes{0.53}{2007/03/21}{topcapt support.}
\g@addto@macro\nag@captions{,caption,captionabove,captionbelow,hc@caption,topcaption}%

%% \S 3.2
\NotAnEnvironment{appendix}%
%% In the same vein:
\@for\sectioning:=frontmatter,mainmatter,backmatter\do{%
  \expandafter\NotAnEnvironment\expandafter{\sectioning}%
}
%% \S 3.3
%% It’s more trouble than it’s worth to have another warning for
%% align*, since it passes through align.
\ObsoleteEnv{eqnarray}{amsmath’s align}
%% \S 3.4 -- nothing to be done --
```

1.5 nag-orthodox.cfg

`nag-orthodox.cfg` warns about usage that is not technically incorrect, but will mostly do things an unwary user may not expect. This includes in particular the usage of font size and style switches as environments (line spacing will be off if the environment does not contain a trailing \par, spurious spaces might occur since the switches don’t \ignorespaces), and, conversely, the usage of center etc. environments as unclosed switches. (Detection of the latter might still be somewhat brittle.)

```
\ProvidesFile{nag-orthodox.cfg}
             [2006/04/19 v1.8 strict rules for nag.sty (ulmi)]
\@for\fontcmd:=tiny,small,footnotesize,normalsize,large,Large,%
               LARGE,huge,Huge\do{%
  \expandafter\NotAnEnvironment\expandafter{\fontcmd}%
}%
\@for\fontcmd:=sffamily,rmfamily,ttfamily,%
               bfseries,mdseries,scshape,%
               itshape,upshape\do{%
  \expandafter\NotAnEnvironment\expandafter{\fontcmd}%
}%
\@for\justsw:=centering,raggedleft,raggedright,%
              RaggedLeft,RaggedRight\do{%
  \expandafter\NotAnEnvironment\expandafter{\justsw}%
}
\@for\justenv:=center,flushleft,flushright\do{%
 \expandafter\NotASwitch\expandafter{\justenv}%
}
```

1.6 nag-abort.cfg

Requesting this nag file will turn all complaints into errors.

```
\ProvidesFile{nag-abort.cfg}
             [2007/11/10 v0.2 treat complaints as errors (ulmi)]
\DeclareRobustCommand\nag@warn[1]{%
  \addtocounter{nag@sins}{1}%
  \PackageError{nag}{#1}{#1}%
}
\DeclareRobustCommand\nag@warnNoLine[1]{%
  \addtocounter{nag@sins}{1}%
  \PackageError{nag}{#1}{#1}%
}
```

1.7 nag-experimental.cfg

Functionality that needs more testing.

```
\ProvidesFile{nag-experimental.cfg}
             [2009/07/04 v0.62alpha2 experimental additions to nag (ulmi)]
```

Patch handling of nofiles: suppressed lines give an Info-level message in the logfile now. The message doesn't quite give the original line, but a sanitized version. Reason: otherwise, we might need to execute the setup code #2.

```
\long\def\nag@protected@dontwrite#1#2#3%
  {\write\m@ne{}%
   \def\nag@line{#3}%
   \@onelevel@sanitize\nag@line
   \PackageInfo{nag}{%
    \string\nofiles\space in effect.
    Did not write line \MessageBreak
    `\nag@line'
   }%
   \if@nobreak\ifvmode\nobreak\fi\fi}%

\if@filesw
  \def\nofiles{%
    \@fileswfalse
    \typeout{No auxiliary output files.^^J}%
    \global\let\protected@write=\nag@protected@dontwrite
    \let\makeindex\relax
    \let\makeglossary\relax}
\else
  % already \nofiles.
  \global\let\protected@write=\nag@protected@dontwrite
\fi
```

Amend "no space for a new foo" message to point out eTeX alleviates some problems in that area.

```
\gdef\ch@ck#1#2#3{%
```

```
  \ifnum\count1#1<#2\else
    \errhelp{%
      eTeX has more counters, dimens, etc., maybe
      that will help.
    }
    \errmessage{No room for a new #3}%
\fi}
\def\@testdef #1#2#3{%
  \def\reserved@a{#3}%
  \expandafter \ifx \csname #1@#2\endcsname\reserved@a
  \else
    \@tempswatrue
    \begingroup
    \@onelevel@sanitize\reserved@a
    \expandafter\let\expandafter\nag@tmpb\csname #1@#2\endcsname
    \ifx\nag@tmpb\relax
      \let\nag@tmpb\@empty
    \else
      \@onelevel@sanitize\nag@tmpb
    \fi
    \PackageInfo{nag}{%
      Label ‘#2’ appears to have changed from\MessageBreak
      ‘\nag@tmpb’\MessageBreak
      to ‘\reserved@a’
    }%
    \endgroup
  \fi}
```

Check if a float that may be positioned b is actually small enough for bottomfraction etc.

```
\let\@xa\expandafter
\newif\ifnag@dofloatsizecheck
\newif\ifnag@allfloatpositionsfailed
\newcommand\nag@allfloatsizechecks{}%
\newcommand\nag@onefloatsizecheck[2]{%
  % #1 is size fraction of textheight,
  % #2 is position to say in warning.
  \ifdim \ht\@currbox>#1\textheight
    \@tempdima -#1\textheight
    \advance \@tempdima \ht\@currbox
    \PackageInfo{nag}{Float too large for #2 by \the\@tempdima}%
    % note we do not truncate.
    % also, it’s too late to add "p" now.
  \else
    \nag@allfloatpositionsfailedfalse
  \fi
}
% \@currbox is current float box,
% \@fps is the current list of float specifiers.
\renewcommand\@largefloatcheck{%
```

```
  \ifdim \ht\@currbox>\textheight
    \@tempdima -\textheight
    \advance \@tempdima \ht\@currbox
    \@latex@warning {Float too large for page by \the\@tempdima}%
    \ht\@currbox \textheight
  \fi
  %% the preceding is the original check.
  \nag@dofloatsizechecktrue
  \nag@allfloatpositionsfailedtrue
  \def\nag@allfloatsizechecks{}%
  \@xa\@xa\@xa\@tfor\@xa\@xa\@xa\nag@fltsz@tmp\@xa\@xa\@xa:\@xa\@xa\@xa=\csname @fps\endcsna
    \ifx\nag@fltsz@tmp\relax
      \nag@dofloatsizecheckfalse
    \fi
    \if\nag@fltsz@tmp !
      \nag@dofloatsizecheckfalse
    \else
      \if\nag@fltsz@tmp t
        \g@addto@macro\nag@allfloatsizechecks
          {\nag@onefloatsizecheck{\topfraction}{top of page}}%
      \else
        \if\nag@fltsz@tmp b
          \g@addto@macro\nag@allfloatsizechecks
            {\nag@onefloatsizecheck{\bottomfraction}{bottom of page}}%
        \else
          \if\nag@fltsz@tmp p
            \nag@allfloatpositionsfailedfalse
          \fi
        \fi
      \fi
    \fi
  }%
  \ifnag@dofloatsizecheck
    \nag@allfloatsizechecks
    \ifnag@allfloatpositionsfailed
      \nag@warn{All float specifiers ‘\@fps’ won’t work}%
    \fi
  \fi
}%
```

More experimental code: warning about files that were requested but not there. The really important one would be a check for include (this is just a typeout in the kernel?!). But as it is, we get warnings that point out missing ToC, LoF etc.

```
\def\@input#1{%
  \IfFileExists{#1}{\@@input\@filef@und}{%
  \typeout{No file #1.}
  \@latex@warning{File ‘#1’ not found}
  %{The file ‘#1’ was requested but not found }
  \protected@edef\nag@nofile{File ‘#1’ requested, but not found}%
  \@xa\AtEndDocument\@xa{%
```

```
    \@xa\@latex@info@no@line\@xa{%
      \nag@nofile
    }%
  }%
}}%
%
\def\@input@#1{\InputIfFileExists{#1}{}{%
  \typeout{No file #1.}
  \@latex@warning{File ‘#1’ not found}
  {The file ‘#1’ was requested but not found }
  \edef\nag@nofile{File ‘#1’ requested, but not found}%
  \@xa\AtEndDocument\@xa{%
    \@xa\@latex@info@no@line\@xa{%
      \nag@nofile
    }%
  }%
}}%
%
```

2 Author-side considerations and implementation.

If you are a package or class author and want to extend the range of `nag` (or prevent `nag` from criticizing your macros), please see the description below, in sections 2.2 and following. It is probably wise to group new rules in a seperate nag file: users can request nag files by passing their name as a package parameter, as shown above for the example of l2tabu.

2.1 Low-level tools.

Identify ourselves.

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{nag}[2011/11/25 0.7 warning about old commands (ulmi)]
\let\@xa\expandafter
\let\@nx\noexpand
```

First of all, two counters we need. The first is used to generate running numbers for replacement macros, the latter is stepped for each complaint we have, so that the user gets a frighteningly high number, showing how sinful he or she is.

```
\newcounter{nag@c}
\renewcommand\thenag@c{\roman{nag@c}}%
\setcounter{nag@c}{1}%
\begingroup
  \let\@addtoreset\@gobbletwo
  \newcounter{nag@sins}%
\endgroup
```

`\nag@prepend` `\nag@prepend{`⟨*cs*⟩`}{`⟨*something*⟩`}`: Prepend ⟨*something*⟩ to the macro definition of \⟨*cs*⟩.

In reality, we do call indirection: save old macro away, redefine macro to do the something, call old macro. (With thanks to Juergen Goebel, Heiko Oberdiek and Rolf Niepraschk (savesym))

From 0.60α_2 on, nag is more robust about not defining commands that are not there. Now, they're not even relaxed.

```
\newcommand\nag@ifundefined[1]{%
  \begingroup
  \@ifundefined{#1}{\endgroup\@firstoftwo}{\endgroup\@secondoftwo}%
}
```

Don't define the macro if it's not there. This confuses caption, which loads ragged2e AtBeginDocument, at which point, RaggedLeft et al. were already defined by us. ...but *do* log a message.

```
\newcommand\nag@prepend[2]{%
  \nag@ifundefined{#1}{%
    % if it doesn't exist, don't do anything.
    \PackageInfo{nag}{%
      Command \@backslashchar#1\space not defined, skipping amendment%
    }%
  }{%
    \nag@ifundefined{#1 }{%
      \let\nag@maybespace\@empty
    }{%
      \let\nag@maybespace\space
      %\PackageInfo{nag}{%
      %  Command \@backslashchar#1\space appears robust\MessageBreak
      %  Modifying `\@backslashchar#1\space' instead.
      %}%
    }%
    \@xa\let
    \csname nag@@#1@\thenag@c\@xa\endcsname
    \csname #1\nag@maybespace\endcsname
    \@xa\DeclareRobustCommand\csname nag@@warning@\thenag@c\@xa\endcsname{%
      #2%
    }%
    \@xa\nag@pr@p@nd\csname #1\nag@maybespace\@xa\endcsname
    \csname nag@@#1@\thenag@c\@xa\endcsname
    \csname nag@@warning@\thenag@c\@xa\endcsname
```

Fun with scoping: one might think we can get away with a (non-local) `\advance\c@nag@c 1\relax` here. This would lead to less hashtable usage. Problem: if a nag@@foo@17 macro ever escapes its scope, it might be bound to something else entirely. This might occur with some of the fancier table packages which use external files?

```
    \stepcounter{nag@c}%
  }%
}
```

```
\newcommand\nag@pr@p@nd[3]{%
  \def#1{#3#2}%
}
```

`\nag@warn` All complaints to the user run through one of these two macros, with or without source line.

```
\DeclareRobustCommand\nag@warn{%
  \addtocounter{nag@sins}{1}%
  \PackageWarning{nag}%
}
\DeclareRobustCommand\nag@warnNoLine{%
  \addtocounter{nag@sins}{1}%
  \PackageWarningNoLine{nag}%
}
\providecommand\PackageInfoNoLine[2]{%
  \PackageInfo{#1}{#2\@gobble}%
}
\DeclareRobustCommand\nag@suggestNoLine[1]{%
  \PackageInfoNoLine{nag}{#1}%
}
```

2.2 Obsoletifying commands.

(No, I do not think that is a proper word either.)

`\ObsoleteCS` Usage: `\ObsoleteCS[`⟨*reason*⟩`]{`⟨*CS*⟩`}{`⟨*suggestions*⟩`}` Mark \⟨*CS*⟩ as obsolete. ⟨*reason*⟩ defaults to obsolete. When the macro is used anyway, the following warning is logged:
`Command \`⟨*CS*⟩` is `⟨*reason*⟩`. Use `⟨*suggestions*⟩` instead.`

```
\newcommand\ObsoleteCS[3][obsolete]{%
  \AtBeginDocument{%
    \nag@prepend{#2}{%
      \nag@warn{%
Command \@backslashchar#2 is #1.
\MessageBreak
Use #3 instead}%
    }%
  }%
}
```

`\ObsoleteEnv`

```
\newcommand\ObsoleteEnv[3][obsolete]{%
  \AtBeginDocument{%
    \nag@prepend{#2}{%
      \nag@warn{%
Environment #2 is #1.
\MessageBreak
Use #3 instead}%
    }%
  }%
}
```

2.3 Obsoletifying packages and classes.

Checking for packages and classes is done by looking for `ver@foo.sty`, which holds the version information that is also displayed by `\listfiles`. This means that we're out of luck if fontenc ever becomes obsolete, because that won't be detected.

First, define a macro to check if a control sequence is defined. Unlike `\@ifundefined`, this will not define the control sequence to `\relax`, but the arguments will be executed in a group. For our purposes, this doesn't matter, because we only give a warning (and `\addtocounter` already is `\global`).

```
\newcommand\nag@ifcsname[3]{%
  \begingroup\@ifundefined{#1}{#3}{#2}\endgroup
}
```

Just because we can, use ϵTeX' `\ifcsname` if we can. This bootstrapping gives me a big grin… Note we add an extra group for compatibility with the non-ϵ case.

```
\nag@ifcsname{ifcsname}{%
  \renewcommand*\nag@ifcsname[3]{%
    \begingroup
      % assume it won't be there.
      \let\tmp@a\@secondoftwo
      \ifcsname #1\endcsname
    % It still might be relax from some other test. Thanks to J\"org
    % Sommer for finding this bug.
      \expandafter\ifx\csname #1\endcsname\relax
      \else
        % it's there after all
        \let\tmp@a\@firstoftwo
      \fi
    \fi
    \tmp@a{#2}{#3}%
    \endgroup
  }%
```

This way of escaping the grouping gives me an even bigger grin.

```
  \global\let\nag@ifcsname\nag@ifcsname
}{}
```

`\ObsoletePackage`

Usage: `\ObsoletePackage[`⟨*reason*⟩`]{`⟨*package*⟩`}{`⟨ *alternative*⟩`}`. Mark ⟨*package*⟩ as obsolete. ⟨*reason*⟩ defaults to obsolete. If the ⟨*package*⟩ is used anyway, at the end of the compilation, the following warning will be displayed:

`Package` ⟨*package*⟩ `is` ⟨*reason*⟩`. Use` ⟨*alternative*⟩ `instead.`

```
\newcommand\ObsoletePackage[3][obsolete]{%
  \AtEndDocument{%
% |\@clsextension| is onlypreamble, for some reason.
    \nag@ifcsname{ver@#2.sty}{%
    \nag@warnNoLine{%
```

```
      Package #2 is #1.\MessageBreak
      Use #3 instead}%
    }{}%
  }%
}
```

`\SuggestedPackage` Usage: `\SuggestedPackage[`⟨*reason*⟩`]{`⟨*package*⟩`}`

```
\newcommand\SuggestedPackage[2][might be useful to you]{%
  \AtEndDocument{%
    \nag@ifcsname{ver@#2.sty}{%
     % Attaboy!
    }{%
    \nag@suggestNoLine{%
      Not loaded: Package #2 #1}%
    }%
  }%
}%
```

`\IncompatiblePackages` Usage: `\IncompatiblePackages[`⟨*reason*⟩`]{`⟨*package*⟩`}{`⟨*package*⟩`}{`⟨*hint*⟩`}`

```
\newcommand\IncompatiblePackages[4][are incompatible]{%
  \AtEndDocument{%
    \nag@ifcsname{ver@#2.sty}{%
      \nag@ifcsname{ver@#3.sty}{%
        \nag@warnNoLine{%
          Packages #2 and #3 #1.\MessageBreak
          #4}%
      }{}%
    }{}
  }%
}%
```

`\ObsoleteClass` Usage: `\ObsoleteClass[`⟨*reason*⟩`]{`⟨*class*⟩`}{`⟨ *alternative*⟩`}`. Mark ⟨*class*⟩ as obsolete. ⟨*reason*⟩ defaults to obsolete. If the ⟨*class*⟩ is used anyway, at the end of the compilation, the following warning will be displayed:
`Class` ⟨*class*⟩ `is` ⟨*reason*⟩`. Use` ⟨*alternative*⟩ `instead.`

```
\newcommand\ObsoleteClass[3][obsolete]{%
  \AtEndDocument{%
% |\@clsextension| is onlypreamble, for some reason.
    \nag@ifcsname{ver@#2.cls}{%
    \nag@warnNoLine{%
      Class #2 is #1.\MessageBreak
      Use #3 instead}%
    }{}%
  }%
}
```

`\BadFileLoadOrder`

```
\def\nag@quark{\nag@quark}
\ifx\@listfiles\@undefined
```

```
  % emulate a silent listfiles
  \def\@listfiles#1\@@{}%
\fi
\newcommand\BadFileLoadOrder[3][This might cause problems]{%
  \AtEndDocument{%
    \nag@ifLoadOrder{#2}{#3}{%
      \nag@warnNoLine{%
        ‘#3’ loaded after ‘#2’.\MessageBreak
        #1}%
    }%
  }%
}
\def\nag@ifLoadOrder#1#2{%
  \def\nag@tmporder@a ##1#1##2\relax{%
    \ifx\nag@quark##2\nag@quark
      \noexpand\@gobble
    \else
      \nag@tmporder@b ##2,#2\relax
    \fi
  }%
  \def\nag@tmporder@b ##1#2##2\relax{%
    \ifx\nag@quark##2\nag@quark
      \noexpand\@gobble
    \else
      \noexpand\@firstofone
    \fi
  }%
  \@xa\protected@edef\@xa\nag@tmporder\@xa{\@xa\nag@tmporder@a\@filelist,,#1\relax}%
  \nag@tmporder
}
```

2.4 Common float errors and no-nos.

We do the following:

- check for presence of a caption
- check for absence of the center environment
- check that a label comes only after a caption

First of all, we define two ifs to memorize whether we have a label and/or a caption in the float already. Package writers may want to set these manually behind `nag`’s back. In this way, they can suppress possible warnings if they know what they’re doing – we only check at the end of the float environment, which gives them plenty of time to call `\csname nag@haslabeltrue\endcsname` et al. (Thanks to Markus Kohm for pointing out this need.) We initialize `\nag@hascaption` to be true because since 0.60, `\label` always checks if it’s after a caption, even outside of floats.

```
\newif\ifnag@haslabel
```

```
\newif\ifnag@hascaption\nag@hascaptiontrue
```

Now, to the work proper: as of 0.60, it is sufficient to set the label and caption flags to false. `\endcenter` now always checks if it is inside a float (looking at `\@captype`). The label and caption commands are amended only once. This should be sufficient: captions are not handled by letting `\caption` to the proper command upon float entry, so we assume nobody redefines `\caption` at runtime, or they provide more entries to `\nag@captions`. Similar for `\label`, and we do not care about the flag setting outside of floats.

```
\newcommand\nag@hackfloat[1]{%
  \nag@prepend{#1}{%
    \global\nag@haslabelfalse\global\nag@hascaptionfalse
  }%
  \nag@prepend{end#1}{%
    \ifnag@hascaption\relax\else
      \nag@warn%
  {#1 with no \protect\caption}%
    \fi
    % labels outside floats shouldn’t complain:
    \global\nag@hascaptiontrue
    % (we do this always because it needs to be global)
  }%
}
```

Add checks to all macros named by `\nag@labels` and `\nag@captions`, respectively. Scoping of presence-of-caption information: Well, maybe I should do it the way the kernel does, which means a label is just as local as `\refstepcounter`'s `\@currentlabel` information as of v0.4. I think we can leave captions global. Big old hack: we do this at `\@preamblecmds`-time, which is after `\AtBeginDocument`, since hyperref loads nameref ABD, and nameref steps all over label. *Note:* We cannot use `\nag@prepend` for this, since it would break the pkgindoc package, which nobody has ever heard of, but it's in the kernel and relies on certain tokens being present in the expansion of `\@preamblecmds`. Now, you pretty much cannot get any later than this.

Note: we cannot exchange the order of the for loops here: if a cs generates both a label and a caption, it shouldn't get complained about.

```
\AtBeginDocument{%
\g@addto@macro{\@preamblecmds}{%
  \@for\labelprovider:=\nag@labels\do{%
    \ifx\labelprovider\@empty\else
    \nag@prepend{\labelprovider}%
        {\nag@captioncheck\nag@haslabeltrue}%
    \fi
  }%
  \@for\captionprovider:=\nag@captions\do{%
    \ifx\captionprovider\@empty\else
      \nag@prepend{\captionprovider}{\global\nag@hascaptiontrue}%
    \fi
  }%
}%
}
\newcommand\nag@captioncheck{%
  \ifnag@hascaption\else
  \nag@warn{\protect\label\space in float, but not after
    \protect\caption}%
  \fi
}
```

Define the lists of commands that are floats, generate labels, and generate captions, respectively. We don't start with defined floats (that is for nag-l2tabu.cfg to set up). Since v0.52, we handle an empty name, so the lists may be empty. Also, no labels and captions are provided by default since v0.52. This has been moved to nag-l2tabu.cfg. See also `\NagDeclareFloat`, which is the user-level wrapper for new floats. Since there are no packages to define new caption or label commands on an user level, there is no wrapper for those.

```
\def\nag@floats{}
\def\nag@labels{}
\def\nag@captions{}
```

We call the above for each float environment named via `\nag@floats`:

```
\newcommand\nag@floatsetup{%
  \@for\flo:=\nag@floats\do{%
    \ifx\flo\@empty\else
      \@xa\nag@hackfloat\@xa{\flo}%
    \fi
  }%
}
```

but only after all other packages get their chance to add to the list:

```
\AtBeginDocument{%
  \nag@floatsetup
}
```

At the very end, we will display a running total of complaints.

```
\AtBeginDocument{%
  \AtEndDocument{%
    \ifnum\value{nag@sins}>0%
    \PackageWarningNoLine{nag}{\arabic{nag@sins} complaints
      in total}%
    \else
    \typeout{No complaints by nag.}%
    \fi
  }%
}
```

3 Switch vs. Environment

People often use switches as environments and vice versa. This is dangerous in because it tends to *almost* work. (Consider font size commands in particular, but also `\centering` vs. `center` environment.) As usual, "it's not an error if you know what you're doing". In particular, it is perfectly valid code to use the `\foo`...`\endfoo` syntax. So, `\NotASwitch` needs to trace the calls to `\foo` and see if they match with corresponding `\endfoo`s with its own stack. This might still be brittle. Fortunately, it is currently only needed for nag-orthodox, where it checks for the justification environments.

First of all, a helper macro we hinge upon:

```
\DeclareRobustCommand\nag@ifCurrentEnvironment[3]{%
  \bgroup
  \def\tmp@a{#1}%
  \ifx\@currenvir\tmp@a
    #2%
  \else
    #3%
  \fi
  \egroup
}
```

And now, the two variations there are:

`\NotAnEnvironment`

Usage:`\NotAnEnvironment{`⟨*command*⟩`}` Issue an error if the user calls `\begin{command}` and not `\command` directly.

```
\newcommand\NotAnEnvironment[1]{%
  \AtBeginDocument{%
    \nag@prepend{#1}{%
      \nag@ifCurrentEnvironment{#1}{%
        \nag@warn{%
          There is no environment ‘‘#1’’.\MessageBreak
          Maybe you want a grouped \@backslashchar#1
        }%
      }{% OK case.
      }%
    }%
  }%
}
```

`\NotASwitch` is a bit more involved:

`\NotASwitch`

Usage:`\NotASwitch{`⟨*command*⟩`}` Issue an error if the user calls `\command` and not `\begin{command}` and mis-nests calls or doesn't call `\endcommand` at all.

```
% we need to maintain a stack of environments that are used in the
% \foo...\endfoo way.
\newcommand\nag@envstack{\relax}

\DeclareRobustCommand\nag@beginenv[1]{%
  % push a begin-entry onto the stack. Form is
  % |{\foo{lineno}}| for environment foo.
  \bgroup
    \@xa\toks@\@xa{\nag@envstack}%
    \xdef\nag@envstack{%
    \@nx{%
    \@xa\@nx\csname #1\endcsname
      \@nx{\the\inputlineno\@nx}%
    \@nx}%
    \the\toks@
    }%
  \egroup
}
\DeclareRobustCommand\nag@endenv[1]{%
  % extract the first entry.
  \@xa\nag@end@nv\nag@envstack\@nil #1\@nil
}

\def\nag@end@nv#1#2\@nil #3\@nil{%
  \def\tmp@a{#1}%
  \def\tmp@b{\relax}%
  \ifx\tmp@a\tmp@b
  % This was the end-of-stack flag.
  \nag@warn{‘‘\@backslashchar end#3’’ without matching
    ‘‘\@backslashchar #3’’}
  \else
    % We may assume this is a proper entry. See if the begin-token on
    % the stack matches what |\nag@endenv| was passed.
    \@xa\ifx\csname #3\@xa\endcsname\@firstoftwo #1%
      %OK case, just pop the entry.
      \gdef\nag@envstack{#2}%
    \else
    % error case
    \nag@warn{%
      You cannot close ‘‘\@xa\string\@firstoftwo #1’’ on line
      \@secondoftwo #1 with ‘‘\@backslashchar end#3’’%
    }%
    % leave it on the stack. Some case of misnesting will always cause
    % horrible amounts of follow-up errors. Also, scare them!
    \fi
  \fi
}
```

At the end, we complain about all the entries that are still on the stack.

```
\AtEndDocument{%
  \@xa\@tfor\@xa\looseends\@xa:\@xa=\nag@envstack\do{%
    \@xa\ifx\looseends\relax\else
      \nag@warnNoLine{Unmatched
        ‘‘\@xa\@xa\@xa\string\@xa\@firstoftwo\looseends’’
        command on line
        \@xa\@xa\@xa\string\@xa\@secondoftwo\looseends%
      }%
    \fi
  }%
}
```

Now, the user-side command is easy.

```
\newcommand\NotASwitch[1]{%
  \AtBeginDocument{%
    \nag@prepend{#1}{%
      \nag@beginenv{#1}%
    }%
    \nag@prepend{end#1}{%
      \nag@endenv{#1}%
    }%
  }%
}
```

4 Compatibility issues

4.1 The **caption** package

Axel Sommerfeldt's `caption` package loads the `ragged2e` package AtBeginDocument (regardless of whether it is needed). This is too late for us to amend the `\RaggedFoo` commands with `\NotAnEnvironment`. Since v0.51 of `nag`, they will then be skipped (with information in the log). Earlier versions would fail because by time `ragged2e` was loaded, the commands were already defined by the amendment process. To make sure the commands *are* amended, load `ragged2e` explicitly yourself.

4.2 The **subfig** package

Starting with v0.52 of `nag`, we recognize the fact that the `\subfloat` command from Steven D. Cochran's `subfig` package is a caption-provider for its fourth argument. Earlier versions would flag use of `\label` as inappropriate. The current implementation works with versions close enough to v1.3 of `subfig`. Since the change is a one-liner, I hope it will be integrated into future versions of `subfig`.

```
\AtBeginDocument{%
  \nag@ifcsname{ver@subfig.sty}{%
    \PackageInfo{nag}{Attempting subfig hack\@gobble}%
    \nag@maybehacksubfig
  }{%
  }%
}
\def\nag@maybehacksubfig{%
  %
  % of course, i need to touch the single longest definition in
  % subfig.sty, to amend one single command...
  %
  % The definition is taken from subfig.sty 1.3 dated 2005/07/05 by
  % S.D. Chochran, where it is called sf@@@subfloat, and appears here
  % under the conditions of section 6 of the LPPL 1.3. The subfig
  % package is available on a CTAN mirror near you.
```

```
%
\long\def\nag@@original@@sf@@@subfloat##1[##2][##3]##4{%
  \@ifundefined{FBsc@max}{%
  }{%
    \FB@readaux{\let\FBsuboheight\relax}%
  }%
  \@tempcnta=\@ne
  \if@minipage
    \@tempcnta=\z@
  \else\ifdim \lastskip=\z@ \else
      \@tempcnta=\tw@
  \fi\fi
  \ifmaincaptiontop
    \sf@top=\sf@nearskip
    \sf@bottom=\sf@farskip
  \else
    \sf@top=\sf@farskip
    \sf@bottom=\sf@nearskip
  \fi
  \leavevmode
  \setbox\@tempboxa \hbox{%
    ##4}%
  \@tempdima=\wd\@tempboxa
  \@ifundefined{FBsc@max}{%
  }{%
    \global\advance\Xhsize-\wd\@tempboxa
    \dimen@=\ht\@tempboxa
    \advance\dimen@\dp\@tempboxa
    \ifdim\dimen@>\FBso@max
    \global\FBso@max\dimen@
    \fi
  }%
  \vtop\bgroup
    \vbox\bgroup
      \ifcase\@tempcnta
      \@minipagefalse
      \or
        \vskip\sf@top
      \or
        \ifdim \lastskip=\z@ \else
          \@tempskipb\sf@top\relax\@xaddvskip
        \fi
      \fi
      \sf@ifpositiontop{%
        \ifx \@empty##3\relax \else
          \sf@subcaption{##1}{##2}{##3}%
          \vskip\sf@capskip
          \vskip\sf@captopadj
        \fi\egroup
        \hrule width0pt height0pt depth0pt
        \box\@tempboxa
      }{%
        \@ifundefined{FBsc@max}{%
          \box\@tempboxa
        }{%
          \ifx\FBsuboheight\relax
            \box\@tempboxa
          \else
            \vbox to \FBsuboheight{\FBafil\box\@tempboxa\FBbfil}%
          \fi}%
        \egroup
      \ifx \@empty##3\relax \else
        \vskip\sf@capskip
        \hrule width0pt height0pt depth0pt
        \sf@subcaption{##1}{##2}{##3}%
      \fi
      }%
      \vskip\sf@bottom
    \egroup
    \@ifundefined{FBsc@max}{%
    }{%
      \addtocounter{FRobj}{-1}%
      \ifnum\c@FRobj=0\else
        \subfloatrowsep
      \fi
    }%
    \ifmaincaptiontop\else
      \global\advance\@nameuse{c@\@captype}\m@ne
    \fi
    \endgroup\ignorespaces}%
%
\expandafter\ifx\csname sf@@@subfloat\endcsname\nag@@original@@sf@@@subfloat
  % yup, that’s it.
  \PackageInfo{nag}{OK, equivalent to subfig 1.3, redefining
    \@backslashchar sf@@@subfloat\@gobble}%
  \global\long\def\sf@@@subfloat##1[##2][##3]##4{%
    \@ifundefined{FBsc@max}{%
  }{%
    \FB@readaux{\let\FBsuboheight\relax}%
  }%
  \@tempcnta=\@ne
  \if@minipage
    \@tempcnta=\z@
  \else\ifdim \lastskip=\z@ \else
      \@tempcnta=\tw@
  \fi\fi
  \ifmaincaptiontop
    \sf@top=\sf@nearskip
    \sf@bottom=\sf@farskip
  \else
```

```
    \sf@top=\sf@farskip
    \sf@bottom=\sf@nearskip
  \fi
  \leavevmode
  \setbox\@tempboxa \hbox{%
    %% ulmi: new 2007/02/25: #4 may contain label command
    \csname nag@hascaptiontrue\endcsname
    %% and that was it.
    ##4}%
  \@tempdima=\wd\@tempboxa
  \@ifundefined{FBsc@max}{%
  }{%
    \global\advance\Xhsize-\wd\@tempboxa
    \dimen@=\ht\@tempboxa
    \advance\dimen@\dp\@tempboxa
    \ifdim\dimen@>\FBso@max
    \global\FBso@max\dimen@
    \fi
  }%
  \vtop\bgroup
    %% ulmi: new 2007/05/10: #2, #3 may contain label command
    \csname nag@hascaptiontrue\endcsname
    %% and that was it.
    \vbox\bgroup
      \ifcase\@tempcnta
      \@minipagefalse
      \or
        \vskip\sf@top
      \or
        \ifdim \lastskip=\z@ \else
          \@tempskipb\sf@top\relax\@xaddvskip
        \fi
      \fi
      \sf@ifpositiontop{%
        \ifx \@empty##3\relax \else
          \sf@subcaption{##1}{##2}{##3}%
          \vskip\sf@capskip
          \vskip\sf@captopadj
        \fi\egroup
        \hrule width0pt height0pt depth0pt
        \box\@tempboxa
      }{%
        \@ifundefined{FBsc@max}{%
          \box\@tempboxa
        }{%
          \ifx\FBsuboheight\relax
            \box\@tempboxa
          \else
            \vbox to \FBsuboheight{\FBafil\box\@tempboxa\FBbfil}%
          \fi}%
```

```
     \egroup
    \ifx \@empty##3\relax \else
     \vskip\sf@capskip
     \hrule width0pt height0pt depth0pt
     \sf@subcaption{##1}{##2}{##3}%
    \fi
    }%
    \vskip\sf@bottom
   \egroup
   \@ifundefined{FBsc@max}{%
   }{%
    \addtocounter{FRobj}{-1}%
    \ifnum\c@FRobj=0\else
     \subfloatrowsep
    \fi
   }%
   \ifmaincaptiontop\else
    \global\advance\@nameuse{c@\@captype}\m@ne
   \fi
   \endgroup\ignorespaces}%
  \else
    \PackageInfo{nag}{Not redefining sf@@@subfloat, it looks odd\@gobble}
  \fi
}
```

4.3 The **float** package

Sorry, there is no way for `nag` to automatically add new float types to check them for captions. However, since v0.52, there is an user-level command `\NagDeclareFloat` that will do the bookkeeping for you, i.e. after your call to `\newfloat`, you call `\NagDeclareFloat` with the first argument to `\newfloat`.

```
\newcommand*\NagDeclareFloat[1]{\g@addto@macro\nag@floats{,#1}}
```

4.4 The **topcapt** package and the **subfig** package

nagdemo exhibits an error when topcapt and subfig are used together, i.e. subfig thinks the caption has not been stepped already. This is not a bug in `nag`.

4.5 The **rotating** package

rotating uses `\centerline` to place rotated floats. As far as I can see, the usage is legitimate there, and using `\centering` instead would change behaviour when the float's dimension are larger than the text body. (Currently, the height of the figure may exceed `\textwidth` without warning.) If this bothers you, go read the warning on p. 3 again.

4.6 Version control packages

Common version control systems like rcs, cvs, svn insert their keywords between dollar signs. Packages that parse these keywords define their commands and usually assume catcode 3, which is not true if either onlyamsmath or nag is loaded. Special handling is introduced for rcs and svninfo. In case of rcsinfo, svn and pgf (yes, it's got internal VC handling that fails when `\pgfuselibrary` is used outside the preamble – thanks to Ralf Thöle for spotting this one), dollar checking is disabled.

5 Loading extensions

Finally, we deal with package options. This is simple: just try to input appropriate nag files.

```
\DeclareOption*{%
  \InputIfFileExists{nag-\CurrentOption.cfg}{%
    \PackageInfo{nag}{%
      Loaded nag-\CurrentOption.cfg
    }
  }{%
    \InputIfFileExists{\CurrentOption.nag}{%
      \PackageWarningNoLine{nag}{%
        Loaded old-style config file \CurrentOption.nag.\MessageBreak
        Consider renaming the file to nag-\CurrentOption.cfg
      }%
    }{%
      \PackageWarningNoLine{nag}{Required ruleset
        \CurrentOption, and it wasn't there}
    }%
  }
}
\ProcessOptions*
```

Change History

0.1
- General: First official version. 1

0.2
- General: Added abort.nag, suggested by Michael Zedler 1
  - Rephrased umlaut.sty warning, suggested by Patrick Happel. . . 1

0.3
- General: Fixed missing globals . . 15
  - New ifdefined that won't relax the commands 1

0.4
- General: bugfix 15
  - config file names changed to free extension 23
  - Handling command vs. environment; bugfixes 1

0.5
- General: Handle the case that somebody else relaxes the ver@-commands. Stack-based NotASwitch. 1

0.51
- `\nag@prepend`: bugfix 11

0.52
- General: Command NagDeclareFloat added 23
  - made eTeX-ifcsname more robust 13
  - twiddle subfig's bowels 19
- `\nag@prepend`: info 11

0.53
- General: bugfix: more Robustness. (Jörg Sommer) 16

0.54
- `\NotASwitch`: bugfix: can't get around the token register. (Jörg Sommer) 17

0.55
- General: Some spaces crept in in 0.5 1

0.60
- General: @preamblecmds 15
  - fixes double-dollar in conjunction with hyperref; documents incompatiblity with rotating. . . . 1
  - Captions/Labels now done only once, and not every time we enter a float 15

0.60alpha
- General: changes the way label/caption is handled, this eliminates the current limit of some thousand floats you can have in your document. (I wonder why nobody noticed). 1

0.60alpha2
- General: is more careful around commands that aren't there. . . 1
- `\nag@prepend`: don't even relax unknown commands (J.Sommer) 11

0.60alpha4
- General: handles eqnarray itself and has code in nag-experimental.cfg to handle double-dollar in a more robust way that onlyamsmath. 1
  - tarballs now unpack into a subdirectory like proper citizens should. 1

0.60alpha5
- General: improves compatibility with subfig. 1

0.61
- General: is 0.61alpha5 with some typos in the docs fixed. 1

0.61alpha1
- General: fixes warnings in toc/lof/lot and unsightly uppercasing. 1
  - roman counter (external file issue) 10
- `\nag@prepend`: Extra indirection of warnings for robustness (uppercasing/LoF issues) 11
- `\nag@warn`: Made robust. 11

0.61alpha2
- General: fixes the warnings, without generating too many duplicates. 1
- `\nag@prepend`: Creep under existing robust cover 11

0.61alpha3
- General: warns about inputs that fail (in particular includes that fail) and notes if a float has position t/b but is too large to ever go into such a position (log only). 1

0.61alpha4
- General: exempts the complaints counter from include trickery. (Previously, nag would get confused if you includeonly only some chapters.) 1
  - sin counter should not be saved by include 10

0.61alpha5
- General: introduces compatibility hacks with version control packages which rely on dollar having constant catcode. (Workaround for svninfo and rcs, all other packages now disable double-dollar checking.) 1

0.61alpha6
- General: Compatibility w/ VCS packages, pgf 4

0.62
- General: fixes a bug in the float

placement code and adds more compatibility with the caption package. 1

0.621
General: Bugfix concerning unknown command in math mode 1

0.622
General: Bugfix: math in captions catcode issue 1

0.62alpha1
General: Bigger warning if all float positions fail 8

0.62alpha2
General: Fix for marginpar etc which don't have fps 8

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.