The `ifoption` package

Michael J. Downes
American Mathematical Society

Version 1.02, 2002/03/04

1 Introduction

The `ifoption` package provides an `\IfOption` command that has certain advantages over the `\@ifpackagewith` command—e.g., default options specified with `\ExecuteOptions` test as true rather than false. In order for this to work properly, mutually exclusive options should be specified with `\DeclareExclusiveOptions`, another command defined by the `ifoption` package.

2 Implementation

2.1 Package info

Standard declaration of package name and date.

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{ifoption}[2002/03/04 v1.02]
```

2.2 Some utility functions

```
\let\@xp\expandafter \let\@nx\noexpand

\def\@True{00}
\def\@False{01}
```

2.3 Background

See the definitions of `\@onefilewithoptions` and `\@pushfilename` in the LaTeX kernel.

When we have

```
\DeclareOption{foo}{some code}
```

in a file named `bar.sty`, LaTeX defines `\ds@foo` to contain *some code*. Then if the package is invoked with

```
\usepackage[foo]{bar}
```

there are two consequences: `foo` is added to the options list for package `bar`, which is the control sequence `\opt@bar.sty`; and when `\ProcessOptions` is called, it executes `\ds@option-name` for each option found in the list. When it is finished running the options, `\ProcessOptions` undefines all the `\ds@whatever` for the current package, which are listed in the control sequence

\@declaredoptions. At the end of bar.sty, \@declaredoptions is globally reset to empty. Thus subordinate packages with options cannot be loaded in the "first half" of a package that also has options, where first half means the area leading up to the \ProcessOptions call.

For mutually exclusive options a, b, c, when option b is invoked it should remove all of its sibling options from the options-actually-used list and add itself to the list. For this task we make use of the fact that \ProcessOptions ends by undefining all the declared options of the current package or documentclass.

```
\def\CurrentPackage{\@currname}
\let\CurrentClass\CurrentPackage

\newcommand{\IfOption}{}
\def\IfPackageOption{\@ifpackagewith}
\def\IfClassOption{\@ifclasswith}
\def\IfOption{%
  \ifx\@currext\@pkgextension \@xp\IfPackageOption
  \else \@xp\IfClassOption
  \fi
  \@currname
}

\newcommand{\DeclareExclusiveOptions}[1]{%
  \xdef\@declaredoptions{\@declaredoptions,#1}%
  \gdef\ProcessExclusiveOptions{\relax}%
  \@for\CurrOption:=#1\do{%
    \@xp\deo@a\csname ds@\CurrOption\@xp\endcsname\@xp{\CurrOption}{#1}%
  }%
}
\def\DeclareBooleanOption{\DeclareExclusiveOptions}

\def\deo@a#1#2#3{%
  \def#1{%
    \g@addto@macro\ProcessExclusiveOptions{\OptionsFalseTrue{#3}{#2}}%
  }%
}

\def\cull@options#1,{%
  \@xp\ifx\csname ds@#1\@gobble\@percentchar\@empty\endcsname\@False
  \else #1,\fi
  \cull@options
}


\newcommand{\OptionsFalseTrue}[2]{%
  \begingroup
  \@for\CurrentOption:=#1\do{%
    \@xp\let\csname ds@\CurrentOption\endcsname\@False
  }%
  \let\ds@\@False
  \@xp\xdef\csname opt@\@currname.\@currext\endcsname{%
    \@xp\@xp\@xp\cull@options\csname opt@\@currname.\@currext\endcsname
```

Lacking \@secondofthree and not keen on defining it ...

```
      ,\@firstoftwo\@firstoftwo,#2%
    }%
    \endgroup
  }
```

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

```
\endinput
```