Typesafe cross-referencing with `typedref`

James Ashton Gregory Seidman

March 26, 2013

Abstract

The `typedref` package replaces `\ref` with `\figureref`, `\sectionref`, `\eqref`, etc. so that you're forced to specify the kind of label you're using. Each reference command also generates appropriate text automatically so that instead of typing '`See Figure~\ref{figure:key}`' only '`See \figureref{key}`' is necessary. The `\label` command is redefined so that it records the type of label with the key. In this way each type of label has its own name-space.

1 Introduction

This package is designed to help avoid cross-referencing errors. In a large work it can be difficult to keep track of the keys used with LaTeX's automatic cross-referencing. It's possible for the output to say 'as in Corollary 3' when in fact there is no 'Corollary 3' but instead 'Lemma 3'. Also it can be taxing to come up with different keys when the keys used for every label share the same name-space. The `typedref` package gives every type of label its own key name-space and defines a separate variant of `\ref` to access each type of label. Use of `\ref` is an error.

2 History

This package is modified from James Ashton's `saferef`, which was released on February 2, 1997. It has been renamed, updated for compatibility with `hyperref`, fixed for use with appendices in non-book documents, and extended to optionally support the Oxford comma by Gregory Seidman. The first release was version 1.0 of `typedref`, as opposed to `saferef`, and dated September 10, 2001. This release is version 1.1, dated March 26, 2013, and adds the `oxfordcomma` package option. The documentation has changed minimally.

3 Compatibility with other packages

The `typedref` package takes care to be compatible with other packages as much as possible. Where commands are redefined, the new definition is in terms of the existing definition where possible so that new features can be added without disrupting existing ones. For this strategy to work, `typedref` must be read *after* other packages which may interact with the features it provides. In particular `typedref` works with AMS-LaTeX 1.1 and 1.2, with the `theorem` package, and with the `hyperref` package when read after them in the preamble.

4 User interface

oxfordcomma The `oxfordcomma` option changes the output of lists of more than two references. The Oxford style includes a comma before “and” in lists, i.e. “one, two, and three” rather than “one, two and three”. With the `oxfordcomma` option set the comma is included; without it the comma is omitted (which has been the default behavior in previous versions of this package).

\label The `\label` command is used exactly as in plain LaTeX, i.e., a single key is supplied as the only argument. Its definition has been changed so that the key is modified by prepending the name of the type of label followed by a colon. If the label follows a `\chapter` command then `\label{key}` will behave as `\label{chapter:key}` does in plain LaTeX.
The types of things that can be labelled each have their own counter and, in general, the name of the counter is used as the name of the label type, i.e., chapters have the label type `chapter` because that is the name of the LaTeX counter used to number them. There are exceptions to this rule however:

- Subsections and subsubsections are both of type `section` since their numbers usually include the section number so that it’s possible to tell what level of section is being referenced by the number itself.
- Subparagraphs are of type `paragraph`.
- Every level of list item is of type `item`, i.e., items numbered using any of the counters `enumi`, `enumii`, `enumiii` or `enumiv`, are all of type `item`.
- Minipage footnotes use the `mpfootnote` counter but are of type `footnote`.
- Of course, appendices use either the `chapter` (if `\chapter` is defined by your document class) or `section` counter but (after the `\appendix` command is executed) they are of type `appendix`.

\pageref The `\pageref` command is not modified by `typedref` but the key used must be modified to match the new definition of `\label` described above.

\ref The `\ref` command is withdrawn so that using it will cause an error. It is replaced by the commands described below.

\appendixref
\chapterref
\figureref
\footnoteref
\itemref
\partref
\sectionref
\tableref

Each of these commands is used to make a cross-reference to a particular kind of label. In each case exactly one argument is required which should be a comma separated list of the label keys to be referenced. These keys are modified by prepending the label type followed by a colon so that they will work with the modified `\label` command.
These commands also provide the text describing the type of label being referenced. The input:

```
See \chapterref{first}.
See \chapterref{first,second}.
See \chapterref{first,second,third}.
```

is equivalent to the following plain LaTeX input:

```
See Chapter~\ref{chapter:first}.
See Chapters \ref{chapter:first} and~\ref{chapter:second}.
See Chapters \ref{chapter:first}, \ref{chapter:second} and~\ref{chapter:third}.
```

(Note that the `oxfordcomma` package option will include a comma before " and" in the last line above.)

`\eqref` `\eqref` is based on the command of the same name provided by AMS-LaTeX. It is used to reference labels of the `equation` type and it produces the equation numbers as displayed with the referenced equation(s). It does not generate the word 'Equation' since this is not usually required.
Note that all of the environments that share the equation numbering should be referenced with `\eqref`. Apart from `equation` and `eqnarray`, this includes the `align`, `gather`, `multline`, etc. environments from AMS-LaTeX and possibly other similar environments from other packages.

`\refname` This command will (re)define a referencing command for a label type. The three arguments required are the label type name, the singular form used to refer to it and the plural form, e.g. `\refname{chapter}{Chapter}{Chapters}`. Note that if a prefix of Equation is desired for equation references, one can give the command `\refname{equation}{Equation}{Equations}` and use `\equationref` instead of `\eqref`.

`\itemname` Sometimes it may be desirable to be more descriptive about items in lists. The `\itemname` command can be used to cause items to be of arbitrary type, e.g., if you were using an `enumerate` environment to list problems, then then `\itemname{problem}` would require them to be referenced like `\problemref{key}`. Of course, it would also be necessary to specify

```
\refname{problem}{Problem}{Problems}
```

for this to work.

`\newtheorem` The `\newtheorem` is redefined so that an extra argument is required. Specify the plural form of the environment's name as an argument immediately following the singular form, e.g.:

```
\newtheorem{theorem}{Theorem}{Theorems}[section]
\newtheorem{lemma}[theorem]{Lemma}{Lemmas}
```

The above would define `\theoremref` and `\lemmaref` and cause labels in the new 'theorem-like' environments to have label types identical to their environment names (`theorem` and `lemma` in this case).

5 An example

The following is a small example which illustrates a few of the features of the package. Just run it through LaTeX twice, then use `makeindex` (with the arguments noted in the comment in the file, and finally run LaTeX again.

```
⟨*example⟩
\documentclass{article}
\usepackage{amsmath}
\usepackage{amsthm}
\usepackage[oxfordcomma]{typedref}
\newtheorem{theorem}{Theorem}{Theorems}
\newtheorem{lemma}[theorem]{Lemma}{Lemmas}
\begin{document}
\section{The first section}
\label{zero}
We'll use the following in \theoremref{one}.
\begin{equation}
a = b\label{two}
\end{equation}
\begin{theorem}
\label{one}
Equation \eqref{two} has nothing to do with \eqref{three} or \eqref{four}.
See also \lemmaref{six}.
\begin{align}
a^2&=b^2+c^2\label{three}\\
a&=\sqrt{b^2+c^2-2bc\cos A}\label{four}
\end{align}
\end{theorem}
Having had this pointless theorem, why not try for a lemma of similar
class.
\section{Another section}
\label{five}
\begin{lemma}
\label{six}
\theoremref{one} (in \sectionref{zero}) contains two equations.
While both are simple, \eqref{four} is more complex than \eqref{three}.
\end{lemma}
\section{The final section}
\label{seven}
This document includes \sectionref{zero,five,seven}.
\lemmaref{six} is on page \pageref{lemma:six}.
\end{document}
⟨/example⟩
```

6 The code

This package is titled `typedref` and it requires LaTeX2e to run.

```
⟨*package⟩
```

```
\def\fileversion{1.1}
\def\filedate{2013/3/26}
\def\docdate{2013/3/26}
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{typedref}[\filedate\space\fileversion\space A Safer Cross-referencing package]
\typeout{Package `typedref' \fileversion\space<\filedate>}
```

These definitions had to be moved into `\AtBeginDocument` because of the definitions `hyperref` puts in `\AtBeginDocument`.

```
\AtBeginDocument{%
```

`\eqnarray` doesn't use `\refstepcounter` directly so we have to modify it.

```
  \let\sr@eqnarray=\eqnarray%
  \def\eqnarray{\def\sr@name{\sr@eq}\sr@eqnarray}%
```

Our redefinition of `\label` involves just the obvious prepending of the label type (as saved in `\sr@label`) to the key.

```
  \let\sr@label=\label%
  \def\label#1{\sr@label{\sr@name:#1}}%
```

AMS-LaTeX uses `\ltx@label` internally instead of `\label` so we need to make these identical in case we're using AMS-LaTeX.

```
  \let\ltx@label=\label%
```

It wouldn't be safe if we could just use the old `\ref` command so we remember its definition and then define it to generate an error message.

```
  \let\sr@ref=\ref%
  \def\ref{\@latex@error{\string\ref\space disallowed with the saferefa package.}}%
}
```

The `\refstepcounter` command is used by LaTeXto increment counters that may be referenced. For compatibility with `hyperref`, however, we must save and redefine `\H@refstepcounter` instead of `\refstepcounter`.

```
\@ifpackageloaded{hyperref}{%
```

First we remember its existing definition.

```
  \let\sr@refstepcounter=\H@refstepcounter%
  \def\H@refstepcounter#1{%
```

We call `\sr@nm@counter-name` if it's defined. This is the mechanism used to allow special behaviour for some counters, but also supports the normal behaviour of the other counters.

```
    \csname sr@nm@#1\endcsname%
```

Finally we call the remembered original definition of `\refstepcounter`.

```
    \sr@refstepcounter{#1}}%
}{%
```

This is the same as above, but deals with `\refstepcounter` instead of `\H@refstepcounter`.

```
  \let\sr@refstepcounter=\refstepcounter%
  \def\refstepcounter#1{%
    \csname sr@nm@#1\endcsname%
    \sr@refstepcounter{#1}}%
}
```

The command `\sr@refs` is called by the various `\ref` variants to do the hard work. It determines whether there's more than one key and generates the singular or plural form of the label type name as appropriate. It also handles the first key by prepending the label type and a colon and passing the result to the original definition of `\ref`. Finally, it calls `\sr@rest` if there are more keys to handle.

```
\def\sr@refs<#1,#2>#3#4#5{%
  \ifx\relax#2\relax
    #4~\sr@ref{#3:#1}%
  \else%
    #5 \sr@ref{#3:#1}%
    \sr@rest<#2>{#3}%
  \fi%
}
```

`\sr@rest` handles every key but the first; prepending the label type and a colon to each one and passing each result to the original definition of `\ref`. It also generates any required commas and the word 'and' between the last two references. It will call `\sr@restmore` if `\sr@refs` was passed more than two keys.

```
\def\sr@rest<#1,#2>#3{%
  \ifx\relax#2\relax
    \ and~\sr@ref{#3:#1}%
  \else%
    , \sr@ref{#3:#1}%
    \sr@restmore<#2>{#3}%
  \fi%
}
```

If the `oxfordcomma` option is not set, `\sr@rest` can be called recursively and produce the desired input, thus `\sr@restmore` is just `\sr@rest` again.

```
\let\sr@restmore=\sr@rest
```

If, however, the `oxfordcomma` option is set, `\sr@restmore` must behave slightly differently. Note the comma before the "and".

```
\DeclareOption{oxfordcomma}{%
  \def\sr@restmore<#1,#2>#3{%
    \ifx\relax#2\relax
      ,\ and~\sr@ref{#3:#1}%
    \else%
      , \sr@ref{#3:#1}%
      \sr@restmore<#2>{#3}%
    \fi%
  }%
}
```

Some base definition of `\eqref` is required since the later redefinition will depend on an existing one. If there isn't an existing one we provide one based on the

AMS-LaTeX1.2 version.

```
\ifx\eqref\@undefined
  \def\eqref#1{\textup{\hbox{\m@th\normalfont(\ignorespaces\ref{#1}\unskip\@@italiccorr)}}}
\fi
```

Now we modify `\eqref` to avoid the use of the forbidden `\ref` command

```
\let\sr@eqref=\eqref
\def\eqref#1{{\let\ref=\sr@ref\sr@eqref{equation:#1}}}
```

The following provides the command which maps label type names to the text to appear in references. Both a singular and plural version must be provided.

```
\def\refname#1#2#3{%
  \expandafter\def\csname#1ref\endcsname##1{\sr@refs<##1,>{#1}{#2}{#3}}}
\def\sr@refname#1#2#3{%
  \expandafter\def\csname sr@nm@#1\endcsname{\xdef\sr@name{#1}}%
  \refname{#1}{#2}{#3}}
```

We use the above command to provide names for the standard LaTeX counters.

```
\sr@refname{appendix}{Appendix}{Appendices}
\sr@refname{chapter}{Chapter}{Chapters}
\sr@refname{figure}{Figure}{Figures}
\sr@refname{footnote}{Footnote}{Footnotes}
\sr@refname{item}{Item}{Items}
\sr@refname{paragraph}{Paragraph}{Paragraphs}
\sr@refname{part}{Part}{Parts}
\sr@refname{section}{Section}{Sections}
\sr@refname{table}{Table}{Tables}
\sr@refname{equation}{Equation}{Equations}
```

We now arrange for some label types to have different names from their counters. This is done by defining a command `\sr@nm@counter-name` which is called by `\label` and which will redefine `\sr@name` as required.
'subsection' and 'subsubsection' are renamed 'section'.

```
\def\sr@nm@subsection{\xdef\sr@name{section}}
\def\sr@nm@subsubsection{\xdef\sr@name{section}}
```

'subparagraph' is renamed 'paragraph'.

```
\def\sr@nm@subparagraph{\xdef\sr@name{paragraph}}
```

'mpfootnote' (footnote in a minipage) is renamed 'footnote'.

```
\def\sr@nm@mpfootnote{\xdef\sr@name{footnote}}
```

Add to the definition of `\appendix` so that it causes the `chapter` or `section` label type to be renamed `appendix`.

```
\let\sr@appendix=\appendix
\@ifundefined{chapter}{
\def\appendix{\sr@appendix\def\sr@nm@section{\xdef\sr@name{appendix}}}
}{
\def\appendix{\sr@appendix\def\sr@nm@chapter{\xdef\sr@name{appendix}}}
}
```

We want to be able to have list items called something other than just 'item' so we'll arrange for the command `\sr@item` to be used to specify the label type of items.

```
\def\sr@nm@enumi{\xdef\sr@name{\sr@item}}
\def\sr@nm@enumii{\xdef\sr@name{\sr@item}}
\def\sr@nm@enumiii{\xdef\sr@name{\sr@item}}
\def\sr@nm@enumiv{\xdef\sr@name{\sr@item}}
```

Now we define a command to set `\sr@item` and use it to set the initial value.

```
\def\itemname#1{\def\sr@item{#1}}
\itemname{item}
```

Likewise, we define a command to set `\sr@eq` and use it to set the initial value. Note that this is an experimental and undocumented feature which may be removed or replaced in the future.

```
\def\eqname#1{\def\sr@eq{#1}}
\eqname{equation}
```

We need to redefine `\newtheorem` for two reasons. Firstly we need to know the plural form of the name for use in multiple references. Secondly, where several 'theorem-like' environments are numbered alike, they all use the same counter. We need to hack the generated environment so that it tells us which environment it is even though it increments a different counter.
The redefinition of `\newtheorem` uses the existing definition so it will work with any existing definition that's compatible with the standard LaTeX `\newtheorem`. In particular, it will work with the 'theorem' or 'amsthm' packages—provided, of course, that these are loaded first.

```
\let\sr@newtheorem=\newtheorem
```

The following is based on the `amsthm` package. The routines just parse the various forms of `\newtheorem` with two possible optional argument forms and a starred form (`amsthm` only). Having gathered up the arguments, they call the remembered original `\newtheorem` and then add a call to `\refname` to the newly defined environment.

```
\def\newtheorem{\@ifstar{\sr@xnthm*}{\sr@xnthm\relax}}
%
\def\sr@oparg#1[#2]{\@ifnextchar[{#1}{#1[#2]}}
%
\def\sr@unthm#1#2{\sr@newtheorem*{#1}{#2}}
%
\def\sr@xnthm#1#2{%
  \let\sr@t\relax
  \ifx *#1% Handle the amsthm starred form of \newtheorem
    \def\sr@t{\sr@unthm{#2}}%
  \else
    \def\sr@t{\sr@oparg{\sr@ynthm{#2}}[]}%
  \fi
  \sr@t
}
%
```

```
\def\sr@ynthm#1[#2]#3#4{%
  \ifx\relax#2\relax
    \def\sr@t{\sr@oparg{\sr@xthm{#1}{#3}{#4}}[]}%
    \expandafter\def\csname sr@nm@#1\endcsname{\xdef\sr@name{#1}}
  \else
    \let\sr@t=\relax
    \sr@newtheorem{#1}[#2]{#3}
    \refname{#1}{#3}{#4}
    \expandafter\let\expandafter\sr@th\csname #1\endcsname
    \expandafter\let\csname sr@th@#1\endcsname\sr@th
    \let\sr@th\relax
    \expandafter\def\csname #1\endcsname{
      \expandafter\def\csname sr@nm@#2\endcsname{\xdef\sr@name{#1}}
      \csname sr@th@#1\endcsname
    }
  \fi
  \sr@t
}
%
\def\sr@xthm#1#2#3[#4]{
  \ifx\relax#4\relax
    \sr@newtheorem{#1}{#2}
    \refname{#1}{#2}{#3}
  \else
    \sr@newtheorem{#1}{#2}[#4]
    \refname{#1}{#2}{#3}
  \fi
}
```

Finally, process the options (just `oxfordcomma` as of now).

```
\ProcessOptions\relax
⟨/package⟩
```