Upgrading from the glossary package to the glossaries package

Nicola L.C. Talbot

2020-03-19

Abstract

The purpose of this document is to provide advice if you want to convert a \LaTeX\ document from using the obsolete glossary package to the replacement glossaries package.

Contents

1 Why the Need for a New Package? 2
2 Package Options 3
3 Defining new glossary types 3
4 \texttt{\make\langle\textit{glossary name}\rangle} 5
5 Storing glossary information 5
6 Adding an entry to the glossary 6
6.1 \texttt{\useglosentry} 6
6.2 \texttt{\useGlosentry} 6
6.3 \texttt{\gls} 7
6.4 \texttt{\glossary} 7
7 Acronyms 8
7.1 \texttt{\acrln} and \texttt{\acrsh} 9
7.2 \texttt{\ifacronymfirstuse} 10
7.3 \texttt{\resetacronym} and \texttt{\unsetacronym} 10
8 Displaying the glossary 12
9 Processing Your Document 13
1 Why the Need for a New Package?

The glossary package started out as an example in a tutorial, but I decided that I may as well package it up and upload it to CTAN. Unfortunately it was fairly rigid and unable to adapt well to the wide variation in glossary styles. Users began making requests for enhancements, but with each enhancement the code became more complicated and bugs crept in. Each fix in one place seemed to cause another problem elsewhere. In the end, it was taking up too much of my time to maintain, so I decided to replace it with a much better designed package. With the new glossaries package:

- you can define irregular plurals;
- glossary terms can have an associated symbol in addition to the name and description;
- new glossary styles are much easier to design;
- you can add dictionaries to supply translations for the fixed names used in headings and by some of the glossary styles;
- you can choose\(^1\) between using `makeindex` or `xindy` to sort the glossary. Using `xindy` means that:
  - there is much better support for terms containing accented or non-Latin characters;
  - there is support for non-standard location numbers;
- you don’t need to remember to escape `makeindex`’s special characters as this is done internally;
- hierarchical entries and homographs are supported;\(^2\)
- there is better support for cross-referencing glossary entries;
- acronyms are just another glossary term which helps to maintain consistency;
- different acronym styles are supported.

---

\(^1\) as from v1.17. Previous versions were designed to be used with `makeindex` only
\(^2\) as from v1.17
2 Package Options

When converting a document that currently uses the obsolete \glossary{} package to the replacement \glossaries{} package, it should be fairly obvious that the first thing you need to do is replace \usepackage{glossary} with \usepackage{glossaries}, however some of the package options are different, so you may need to change those as well. Table 1 shows the mappings from the \glossary{} to the \glossaries{} package options.

<table>
<thead>
<tr>
<th>\glossary{} option</th>
<th>\glossaries{} option</th>
</tr>
</thead>
<tbody>
<tr>
<td>style=list</td>
<td>style=list</td>
</tr>
<tr>
<td>style=altlist</td>
<td>style=altlist</td>
</tr>
<tr>
<td>style=long,header=none,border=none,cols=2</td>
<td>style=long,header=none,border=none,cols=2</td>
</tr>
<tr>
<td>style=long,header=plain,border=none,cols=2</td>
<td>style=long,header=plain,border=none,cols=2</td>
</tr>
<tr>
<td>style=long,header=none,border=plain,cols=2</td>
<td>style=long,header=none,border=plain,cols=2</td>
</tr>
<tr>
<td>style=long,header=plain,border=plain,cols=2</td>
<td>style=long,header=plain,border=plain,cols=2</td>
</tr>
<tr>
<td>style=long,header=none,border=none,cols=3</td>
<td>style=long,header=none,border=none,cols=3</td>
</tr>
<tr>
<td>style=long,header=plain,border=none,cols=3</td>
<td>style=long,header=plain,border=none,cols=3</td>
</tr>
<tr>
<td>style=long,header=plain,border=plain,cols=3</td>
<td>style=long,header=plain,border=plain,cols=3</td>
</tr>
<tr>
<td>style=super,header=none,border=none,cols=2</td>
<td>style=super,header=none,border=none,cols=2</td>
</tr>
<tr>
<td>style=super,header=plain,border=none,cols=2</td>
<td>style=super,header=plain,border=none,cols=2</td>
</tr>
<tr>
<td>style=super,header=plain,border=plain,cols=2</td>
<td>style=super,header=plain,border=plain,cols=2</td>
</tr>
<tr>
<td>style=super,header=none,border=none,cols=3</td>
<td>style=super,header=none,border=none,cols=3</td>
</tr>
<tr>
<td>style=super,header=plain,border=none,cols=3</td>
<td>style=super,header=plain,border=none,cols=3</td>
</tr>
<tr>
<td>style=super,header=plain,border=plain,cols=3</td>
<td>style=super,header=plain,border=plain,cols=3</td>
</tr>
<tr>
<td>number=none</td>
<td>number={counter name}</td>
</tr>
<tr>
<td>toc</td>
<td>toc</td>
</tr>
<tr>
<td>hypertoc</td>
<td>hypertoc</td>
</tr>
<tr>
<td>hyper</td>
<td>hyper</td>
</tr>
<tr>
<td>section=true</td>
<td>section=false</td>
</tr>
<tr>
<td>acronym</td>
<td>acronym</td>
</tr>
<tr>
<td>global</td>
<td>global</td>
</tr>
</tbody>
</table>

3 Defining new glossary types

If you have created new glossary types, you will need to replace all instances of
The \langle old style list \rangle optional argument can be converted to \langle new style \rangle using the same mapping given in Table 1.

For example, if your document contains the following:

\begin{verbatim}
\newglossarytype{nlg}{notation}{not}{ntn}{style=long,header}
\newcommand{\notationname}{Index of Notation}
\end{verbatim}

You will need to replace the above two lines with:

\begin{verbatim}
\newglossary[nlg]{notation}{not}{ntn}{Index of Notation}
\end{verbatim}

in the preamble and

\begin{verbatim}
\setglossarystyle{longheader}
\end{verbatim}

immediately prior to displaying this glossary. Alternatively, you can specify the style using the style key in the optional argument of \texttt{printglossary}. For example:

\begin{verbatim}
\printglossary[type=notation,style=longheader]
\end{verbatim}

Note that the glossary title is no longer specified using \langle glossary-type\rangle name (except for \texttt{glossaryname} and \texttt{acronymname}) but is instead specified in the \langle title \rangle argument of \texttt{newglossary}. The short title which is specified in the glossary package by the command \texttt{\short{glossary-name}} name is now specified using the \texttt{toctitle} key in the optional argument to \texttt{printglossary}. 
4 \texttt{\textbackslash make\langle glossary name\rangle}

All instances of \texttt{\textbackslash make\langle glossary name\rangle} (e.g. \texttt{\textbackslash makeglossary} and \texttt{\textbackslash makeacronym}) should be replaced by the single command \texttt{\textbackslash makeglossaries}. For example, if your document contained the following:

\begin{verbatim}
\makeglossary
\makeacronym
\end{verbatim}

then you should replace both lines with the single line:

\begin{verbatim}
\makeglossaries
\end{verbatim}

5 Storing glossary information

With the old glossary package you could optionally store glossary information for later use, or you could simply use \texttt{\textbackslash glossary} whenever you wanted to add information to the glossary. With the new glossaries package, the latter option is no longer available.\footnote{mainly because having a key value list in \texttt{\textbackslash glossary} caused problems, but it also helps consistency.} If you have stored all the glossary information using \texttt{\textbackslash storeglosentry}, then you will need to convert these commands into the equivalent \texttt{\textbackslash newglossaryentry}. If you have only used \texttt{\textbackslash glossary}, then see Section 6.4.

Substitute all instances of

\begin{verbatim}
\storeglosentry\{(label)\}\{(gls-entry)\}
\end{verbatim}

glossary

with

\begin{verbatim}
\newglossaryentry\{(label)\}\{(gls-entry)\}
\end{verbatim}

glossaries

This should be fairly easy to do using the search and replace facility in your editor (but see notes below).

If you have used the optional argument of \texttt{\textbackslash storeglosentry} (i.e. you have multiple glossaries) then you will need to substitute

\begin{verbatim}
\storeglosentry\{(gls-type)\}\{(label)\}\{(gls-entry)\}
\end{verbatim}

glossary

with

\begin{verbatim}
\newglossaryentry\{(label)\}\{(gls-entry)\},type=\{(gls-type)\}
\end{verbatim}

glossaries
The glossary entry information \texttt{⟨gls-entry⟩} may also need changing. If \texttt{⟨gls-entry⟩} contains any of `makeindex`'s special characters (i.e. @ ! " or |) then they should no longer be escaped with " since the `glossaries` package deals with these characters internally. For example, if your document contains the following:

\begin{verbatim}
\storeglosentry{card}{name={$|$mathcal{S}|$},
description={The cardinality of the set {$\mathcal{S}$}}}
\end{verbatim}

then you will need to replace it with:

\begin{verbatim}
\newglossaryentry{card}{name={$|$mathcal{S}|$},
description={The cardinality of the set {$\mathcal{S}$}}}
\end{verbatim}

The format and number keys available in \texttt{\storeglosentry} are not available with \texttt{\newglossaryentry}.

\section{Adding an entry to the glossary}

The glossary package provided two basic means to add information to the glossary: firstly, the term was defined using \texttt{\storeglosentry} and the entries for that term were added using \texttt{\useglosentry}, \texttt{\useGlosentry} and \texttt{\gls}. Secondly, the term was added to the glossary using \texttt{\glossary}. This second approach is unavailable with the `glossaries` package.

\subsection{\texttt{\useglosentry}}

The `glossary` package allows you to add information to the glossary for a predefined term without producing any text in the document using

\begin{verbatim}
\useglosentry[⟨old options⟩]{⟨label⟩}
\end{verbatim}

glossary

Any occurrences of this command will need to be replaced with

\begin{verbatim}
\glsadd[⟨new options⟩]{⟨label⟩}
\end{verbatim}

glossaries

The \texttt{format} key in \texttt{⟨old options⟩} remains the same in \texttt{⟨new options⟩}. However the \texttt{number=⟨counter name⟩} key in \texttt{⟨old options⟩} should be replaced with \texttt{counter=⟨counter name⟩} in \texttt{⟨new options⟩}.

\subsection{\texttt{\useGlosentry}}

The `glossary` package allows you to add information to the glossary for a predefined term with the given text using

\begin{verbatim}
\end{verbatim}
Any occurrences of this command will need to be replaced with

\glslink{new options}{label}{text}

The mapping from \texttt{old options} to \texttt{new options} is the same as that given Section 6.1.

### 6.3 \texttt{\gls}

Both the glossary and the glossaries packages define the command \texttt{\gls}. In this case, the only thing you need to change is the number key in the optional argument to \texttt{counter}. Note that the new form of \texttt{\gls} also takes a final optional argument which can be used to insert text into the automatically generated text.

### 6.4 \texttt{\glossary}

When using the glossaries package, you should not use \texttt{\glossary} directly.\(^4\) If, with the old package, you have opted to explicitly use \texttt{\glossary} instead of storing the glossary information with \texttt{\storeglosentry}, then converting from glossary to glossaries will be more time-consuming, although in the end, I hope you will see the benefits.\(^5\) If you have used \texttt{\glossary} with the old glossary package, you will instead need to define the relevant glossary terms using \texttt{\newglossaryentry} and reference the terms using \texttt{\glsadd}, \texttt{\glslink}, \texttt{\gls} etc.

If you don’t like the idea of continually scrolling back to the preamble to type all your \texttt{\newglossaryentry} commands, you may prefer to create a new file, in which to store all these commands, and then input that file in your document’s preamble. Most text editors and front-ends allow you to have multiple files open, and you can tab back and forth between them.

---

\(^4\)This is because \texttt{\glossary} requires the argument to be in a specific format and doesn’t use the \texttt{(key)=(value)} format that the old glossary package used. The new package’s internal commands set this format, as well as escaping any of \texttt{makeindex’s} or \texttt{xindy’s} special characters. What’s more, the format has changed as from v1.17 to allow the new package to be used with either \texttt{makeindex} or \texttt{xindy}.

\(^5\)From the user’s point of view, using \texttt{\glossary} throughout the document is time consuming, and if you use it more than once for the same term, there’s a chance extra spaces may creep in which will cause \texttt{makeindex} to treat the two entries as different terms, even though they look the same in the document.
7 Acronyms

In the glossary package, acronyms were treated differently to glossary entries. This resulted in inconsistencies and sprawling unmaintainable code. The new glossaries package treats acronyms in exactly the same way as normal glossary terms. In fact, in the glossaries package, the default definition of:

\newacronym[\{options\}]{\{label\}}{\{abbrv\}}{\{long\}}

is a shortcut for:

\newglossaryentry{\{label\}}{type=\texttt{acronymtype},
name=\{\texttt{abbrv}\},
description=\{\texttt{long}\},
text=\{\texttt{abbrv}\},
first=\{\texttt{long} \ (\texttt{abbrv})\},
plural=\{\texttt{abbrv}s\},
firstplural=\{\texttt{long}s \ (\texttt{abbrv}s)\},
\{options\}}

This is different to the glossary package which set the name key to \{\texttt{long} \ (\texttt{abbrv})\} and allowed you to set a description using the description key. If you still want to do this, you can use one of the description styles, such as \texttt{long-short-desc}, and use the description key in the optional argument of \texttt{newacronym}.

For example, if your document originally had the following:

\newacronym{SVM}{Support Vector Machine}{description=Statistical pattern recognition technique}

Then you would need to first set the style:

\setacronymstyle{long-short-desc}

and change the acronym definition to:

\newacronym[description=Statistical pattern recognition technique]{svm}{SVM}{Support Vector Machine}

You can then reference the acronym using any of the new referencing commands, such as \texttt{\gls} or \texttt{\glsadd}.

With the old glossary package, when you defined an acronym, it also defined a command \texttt{\{acr-name\}} which could be used to display the acronym in the text. So the above SVM example would create the command \texttt{\SVM} with the old package. In the new glossaries package, the acronyms are just another type of glossary entry, so they are displayed using \texttt{\gls{\{label\}}}.

Therefore, in the above example, you will also need to replace all occurrences of \texttt{\SVM} with \texttt{\gls{svm}}.
If you have used `\useacronym` instead of `\(acr-name\)`, then you will need to replace all occurrences of

```
\useacronym[\{insert\}]{acr-name}
```

glossary

with

```
\gls[\{label\}][\{insert\}]
```

glossaries

Note that the starred versions of `\useacronym` and `\(acr-name\)` (which make the first letter uppercase) should be replaced with `\Gls[\{label\}]`. Alternatively (as from v1.18 of the `glossaries` package), you can use `\oldacronym` which uses the same syntax as the old `glossary` package's `\newacronym` and also defines `\(acr-name\)`. For example, if your document originally had the following:

```
\newacronym{SVM}{Support Vector Machine}{description=Statistical pattern recognition technique}
```

glossary

then you can change this to:

```
\oldacronym{SVM}{Support Vector Machine}{description=Statistical pattern recognition technique}
```

glossaries

You can then continue to use `\SVM`. However, remember that LaTeX generally ignores spaces after command names that consist of alphabetical characters. You will therefore need to force a space after `\(acr-name\)`, unless you also load the `xspace` package. (See Section 13 of the `glossaries` documentation for further details.) Note that `\oldacronym` uses its first argument to define the acronym’s label (as used by commands like `\gls`), so in the above example, with the new `glossaries` package, `\SVM` becomes a shortcut for `\gls{SVM}` and `\SVM*` becomes a shortcut for `\Gls{SVM}`.

### 7.1 `\acrln` and `\acrsh`

In the `glossary` package, it is possible to produce the long and short forms of an acronym without adding an entry to the glossary using `\acrln` and `\acrsh`. With the `glossaries` package (provided you defined the acronym using `\newacronym` or `\oldacronym` and provided you haven’t redefined `\newacronym`) you can replace

```
\acrsh{acr-name}
```

glossary
with
\acrshort{{label}}
glossaries

and you can replace
\acrln{{acr-name}}
glossary

with
\acrlong{{label}}
glossaries

The glossaries package also provides the related commands \acrshortpl (plural short form) and \acrlongpl (plural long form) as well as upper case variations. If you use the glossaries “shortcuts” package option, you can use \acs in place of \acrshort and \acl in place of \acrlong.

See Section 13 of the glossaries manual for further details of how to use these commands.

7.2 \ifacronymfirstuse
The glossary package command
\ifacronymfirstuse{{acr-name}}{{text1}}{{text2}}
glossary
can be replaced by the glossaries command:
\ifglsused{{label}}{{text2}}{{text1}}
glossaries

Note that \ifglsused evaluates the opposite condition to that of \ifacronymfirstuse which is why the last two arguments have been reversed.

7.3 \resetacronym and \unsetacronym
The glossary package allows you to reset and unset the acronym flag which is used to determine whether the acronym has been used in the document. The glossaries package also provides a means to do this on either a local or a global level. To reset an acronym, you will need to replace:
To unset an acronym, you will need to replace:
\unsetacronym{acr-name} with either
\glsunset{label} or \glslocalunset{label}.

To unset all acronyms, you will need to replace:
\unsetallacronyms with either
\glsunsetall{acronymtype} or \glslocalunsetall{acronymtype}.

To reset all acronyms, you will need to replace:
\resetallacronyms with \glsresetall{acronymtype} or \glslocalresetall{acronymtype}.

To reset an acronym, you will need to replace:
\glsreset{acr-name} with either
\glsreset{label} or \glslocalreset{label}.

To reset all acronyms, you will need to replace:
\resetallacronyms with \glsresetall{acronymtype} or \glslocalresetall{acronymtype}.

To unset all acronyms, you will need to replace:
8 Displaying the glossary

The glossary package provides the command `\printglossary` (or `\print(type)` for other glossary types) which can be used to print individual glossaries. The glossaries package provides the command `\printglossaries` which will print all the glossaries which have been defined, or `\printglossary[⟨options⟩]` to print individual glossaries. So if you just have `\printglossary`, then you can leave it as it is, but if you have, say:

```latex
\printglossary
\printglossary[acronym]
```

or

```latex
\printglossary
\printacronym
```

then you will need to replace this with either

```latex
\printglossaries
```

or

```latex
\printglossary
\printglossary[type=⟨acronymtype⟩]
```

The glossary package allows you to specify a short title (for the table of contents and page header) by defining a command of the form `\short(glossary-type)name`. The glossaries package doesn’t do this, but instead provides the `toctitle` key which can be used in the optional argument to `\printglossary`. For example, if you have created a new glossary type called `notation`, and you had defined

```latex
\newcommand{\shortnotationname}{Notation}
```

```latex
\printglossary
\printglossary[acronym]
```

```latex
\printglossary[acronym]
\printglossary[type=⟨acronymtype⟩]
```
then you would need to use the toctitle key:

\printglossary[type=notation, toctitle=Notation]

The glossaries package will ignore \shortnotationname, so unless you have used it elsewhere in the document, you may as well remove the definition.

9 Processing Your Document

If you convert your document from using the glossary package to the glossaries package, you will need to delete any of the additional files, such as the .glo file, that were created by the glossary package, as the glossaries package uses a different format. Remember also, that if you used the makeglos Perl script, you will need to use the makeglossaries Perl script instead. As from v1.17, the glossaries package can be used with either makeindex or xindy. Since xindy was designed to be multilingual, the new glossaries package is a much better option for non-English documents. If you use the extension package, glossaries-extra, then you also have the option of using bib2gls instead (which also provides multilingual support).

For further information on using makeglossaries, makeindex or xindy to create your glossaries, see Section 1.4 of the glossaries documentation.

10 Troubleshooting

Please check the FAQ\(^6\) for the glossaries package if you have any problems.

Index

<table>
<thead>
<tr>
<th></th>
<th>B</th>
<th>makeglossaries</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>bib2gls</td>
<td>makeindex</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>file types</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>.glo</td>
<td></td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>glossaries-extra package</td>
<td></td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>\gls options</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>counter</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>number</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>\glsadd options</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>counter</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>makeglos</td>
<td></td>
<td>12</td>
</tr>
</tbody>
</table>

\(^6\)http://www.dickimaw-books.com/faqs/glossariesfaq.html
<table>
<thead>
<tr>
<th>Command</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>U</td>
<td>6</td>
</tr>
<tr>
<td>\useglosentry</td>
<td></td>
</tr>
<tr>
<td>options</td>
<td></td>
</tr>
<tr>
<td>format</td>
<td>6</td>
</tr>
<tr>
<td>x</td>
<td></td>
</tr>
<tr>
<td>xindy</td>
<td>1, 7, 12</td>
</tr>
</tbody>
</table>