

Documented Code For glossaries v4.11

Nicola L.C. Talbot

Dickimaw Books

<http://www.dickimaw-books.com/>

2014-09-01

This is the documented code for the glossaries package. This bundle comes with the following documentation:

[glossariesbegin.pdf](#) If you are a complete beginner, start with “The glossaries package: a guide for beginners”.

[glossary2glossaries.pdf](#) If you are moving over from the obsolete glossary package, read “Upgrading from the glossary package to the glossaries package”.

[glossaries-user.pdf](#) For the main user guide, read “glossaries.sty v4.11: L^AT_EX2e Package to Assist Generating Glossaries”.

[mfirstuc-manual.pdf](#) The commands provided by the mfirstuc package are briefly described in “mfirstuc.sty: uppercasing first letter”.

[glossaries-code.pdf](#) This document is for advanced users wishing to know more about the inner workings of the glossaries package.

INSTALL Installation instructions.

CHANGES Change log.

README Package summary.

The user level commands described in the user manual ([glossaries-user.pdf](#)) may be considered “future-proof”. Even if they become deprecated, they should still work for old documents (although they may not work in a document that also contains new commands introduced since the old commands were deprecated, and you may need to specify a compatibility mode).

The internal commands in *this* document that aren't documented in the *user manual* should not be considered future-proof and are liable to change. If you want a new user level command, you can post a feature request at <http://www.dickimaw-books.com/feature-request.html>. If you are a package writer wanting to integrate your package with glossaries, it's better to request a new user level command than to hack these internals.

Contents

1 Main Package Code	4
1.1 Package Definition	4
1.2 Package Options	6
1.3 Default values	30
1.4 Xindy	39
1.5 Loops and conditionals	48
1.6 Defining new glossaries	54
1.7 Defining new entries	58
1.8 Resetting and unsetting entry flags	78
1.9 Loading files containing glossary entries	80
1.10 Using glossary entries in the text	81
1.10.1 Links to glossary entries	91
1.10.2 Displaying entry details without adding information to the glossary	133
1.11 Adding an entry to the glossary without generating text	141
1.12 Creating associated files	142
1.13 Writing information to associated files	158
1.14 Glossary Entry Cross-References	164
1.15 Displaying the glossary	166
1.16 Acronyms	195
1.17 Predefined acronym styles	199
1.18 Predefined Glossary Styles	230
1.19 Debugging Commands	231
1.20 Compatibility with version 2.07 and below	236
2 Prefix Support (glossaries-prefix Code)	237
3 Mfirstuc Documented Code	243
4 Mfirstuc-english Documented Code	246
5 Glossary Styles	246
5.1 Glossary hyper-navigation definitions (glossary-hypernav package)	246
5.2 In-line Style (glossary-inline.sty)	249
5.3 List Style (glossary-list.sty)	251
5.4 Glossary Styles using longtable (the glossary-long package)	254
5.5 Glossary Styles using longtable (the glossary-longragged package)	260
5.6 Glossary Styles using multicol (glossary-mcols.sty)	265
5.7 Glossary Styles using supertabular environment (glossary-super package)	269
5.8 Glossary Styles using supertabular environment (glossary-superragged package)	276
5.9 Tree Styles (glossary-tree.sty)	282

6 glossaries-compatible-207	290
7 Accessibility Support (glossaries-accsupp Code)	310
7.1 Defining Replacement Text	311
7.2 Accessing Replacement Text	314
7.3 Displaying the Glossary	329
7.4 Acronyms	331
7.5 Debugging Commands	345
8 Multi-Lingual Support	346
8.1 Babel Captions	347
8.2 Polyglossia Captions	352
8.3 Brazilian Dictionary	356
8.4 Danish Dictionary	356
8.5 Dutch Dictionary	356
8.6 English Dictionary	357
8.7 French Dictionary	357
8.8 German Dictionary	357
8.9 Irish Dictionary	357
8.10 Italian Dictionary	358
8.11 Magyar Dictionary	358
8.12 Polish Dictionary	358
8.13 Serbian Dictionary	359
8.14 Spanish Dictionary	359
Glossary	359
Change History	359
Index	383

1 Main Package Code

1.1 Package Definition

This package requires $\LaTeX 2\epsilon$.

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries}[2014/09/01 v4.11 (NLCT)]
```

Required packages:

```
3 \RequirePackage{ifthen}
4 \RequirePackage{xkeyval}[2006/11/18]
5 \RequirePackage{mfirstuc}
```

The textcase package has much better case changing handling, so use `\MakeTextUppercase` instead of `\MakeUppercase`

```
6 \RequirePackage{textcase}
```

```
7 \renewcommand*{\mfirstucMakeUppercase}{\MakeTextUppercase}%
```

```
8 \RequirePackage{xfor}
```

```
9 \RequirePackage{datatool-base}
```

Need to use `\new@ifnextchar` instead of `\@ifnextchar` in commands that have a final optional argument (such as `\gls`) so require `.` Thanks to Morten Høgholm for suggesting this. (This has replaced using the `xspace` package.)

```
10 \RequirePackage{amsgen}
```

As from v3.0, now loading `etoolbox`:

```
11 \RequirePackage{etoolbox}
```

Check if doc has been loaded.

```
\if@gls@docloaded
```

```
12 \newif\if@gls@docloaded
```

```
13 \@ifpackageloaded{doc}%
```

```
14 {%
```

```
15   \@gls@docloadedtrue
```

```
16 }%
```

```
17 {%
```

```
18   \@ifclassloaded{nlctdoc}{\@gls@docloadedtrue}{\@gls@docloadedfalse}%
```

```
19 }
```

```
20 \if@gls@docloaded
```

`\doc` has been loaded, so some modifications need to be made to ensure both packages can work together. The amount of conflict has been reduced as from v4.11 and no longer involves patching internal commands.

`\PrintChanges` needs to use doc's version of `theglossary`, so save that.

```
\glsorg@theglossary
```

```
21 \let\glsorg@theglossary\theglossary
```

```
sorg@endtheglossary
```

```
22 \let\glsorg@endtheglossary\endtheglossary
```

`\PrintChanges` Now redefine `\PrintChanges` so that it uses the original `theglossary` environment.

```
23 \let\glsorg@PrintChanges\PrintChanges
```

```
24 \renewcommand{\PrintChanges}{%
```

```
25   \begingroup
```

```
26     \let\theglossary\glsorg@theglossary
```

```
27     \let\endtheglossary\glsorg@endtheglossary
```

```
28     \glsorg@PrintChanges
```

```
29   \endgroup
```

```
30 }
```

End of doc stuff.

```
31 \fi
```

1.2 Package Options

`toc` The `toc` package option will add the glossaries to the table of contents. This is a boolean key, if the value is omitted it is taken to be true.

```
32 \define@boolkey{glossaries.sty}[gls]{toc}[true]{}
```

`numberline` The `numberline` package option adds `\numberline` to `\addcontentsline`. Note that this option only has an effect if used in with `toc=true`.

```
33 \define@boolkey{glossaries.sty}[gls]{numberline}[true]{}
```

`\@glossarysec` The sectional unit used to start the glossary is stored in `\@glossarysec`. If chapters are defined, this is initialised to `chapter`, otherwise it is initialised to `section`.

```
34 \ifcsundef{chapter}%  
35   {\newcommand*\@glossarysec}{section}}%  
36   {\newcommand*\@glossarysec}{chapter}}
```

`section` The `section` key can be used to set the sectional unit. If no unit is specified, use `section` as the default. The starred form of the named sectional unit will be used. If you want some other way to start the glossary section (e.g. a numbered section) you will have to redefined `\glossarysection`.

```
37 \define@choicekey{glossaries.sty}{section}{part,chapter,section,%  
38 subsection,subsubsection,paragraph,subparagraph}[section]{}%  
39   \renewcommand*\@glossarysec{#1}}
```

Determine whether or not to use numbered sections.

`\@glossarysecstar`

```
40 \newcommand*\@glossarysecstar{*}
```

`\@glossaryseclabel`

```
41 \newcommand*\@glossaryseclabel{}
```

`\glsautoprefix` Prefix to add before label if automatically generated:

```
42 \newcommand*\glsautoprefix{}
```

`numberedsection`

```
43 \define@choicekey{glossaries.sty}{numberedsection}[\val\nr]{}%  
44 false,nolabel,autolabel,nameref}[nolabel]{}%  
45   \ifcase\nr\relax  
46     \renewcommand*\@glossarysecstar{*}%  
47     \renewcommand*\@glossaryseclabel{}%  
48   \or  
49     \renewcommand*\@glossarysecstar{}%  
50     \renewcommand*\@glossaryseclabel{}%  
51   \or  
52     \renewcommand*\@glossarysecstar{}%  
53     \renewcommand*\@glossaryseclabel{}%
```

```

54     \label{\glsautoprefix\@glo@type}}%
55   \or
56     \renewcommand*{\@glossarysecstar}{*}%
57     \renewcommand*{\@glossaryseclabel}{%
58       \protected@edef\@currentlabelname{\glossarytoctitle}%
59       \label{\glsautoprefix\@glo@type}}%
60   \fi
61 }

```

The default glossary style is stored in `\@glossary@default@style`. This is initialised to `list`. (The `list` style is defined in the accompanying package described in [subsection 1.18](#).)

`\@glossary@default@style`

```
62 \newcommand*{\@glossary@default@style}{list}
```

`style` The default glossary style can be changed using the `style` package option. The value can be the name of any defined glossary style. The glossary style is set at the beginning of the document, so you can still use the `style` key to set a style that is defined in another package. This package comes with some predefined styles that are defined in [subsection 1.18](#).

```
63 \define@key{glossaries.sty}{style}{%
64   \renewcommand*{\@glossary@default@style}{#1}%
65 }

```

Each `\DeclareOptionX` needs a corresponding `\DeclareOption` so that it can be passed as a document class option, so define a command that will implement both.

`\@gls@declareoption`

```
66 \newcommand*{\@gls@declareoption}[2]{%
67   \DeclareOptionX{#1}{#2}%
68   \DeclareOption{#1}{#2}%
69 }

```

Each entry within a given glossary will have an associated number list. By default, this refers to the page numbers on which that entry has been used, but it can also refer to any counter used in the document (such as the section or equation counters). The default number list format displays the number list “as is”:

`\glossaryentrynumbers`

```
70 \newcommand*{\glossaryentrynumbers}[1]{#1\gls@save@numberlist{#1}}
```

`nonumberlist` Note that the entire number list for a given entry will be passed to `\glossaryentrynumbers` so any font changes will also be applied to the delimiters. The `nonumberlist` package option suppresses the number lists (this simply redefines `\glossaryentrynumbers` to ignore its argument).

```

71 \@gls@declareoption{nonumberlist}{%
72   \renewcommand*{\glossaryentrynumbers}[1]{\gls@save@numberlist{#1}}%
73 }

```

`savenumberlist` Provide means to store the number list for entries.

```

74 \define@boolkey{glossaries.sty}[gls]{savenumberlist}[true]{}
75 \glssavenumberlistfalse

```

`o@seeautonumberlist`

```

76 \newcommand*\@glo@seeautonumberlist{}

```

`seeautonumberlist` Automatically activates number list for entries containing the see key.

```

77 \@gls@declareoption{seeautonumberlist}{%
78   \renewcommand*{\@glo@seeautonumberlist}{%
79     \def\@glo@prefix{\glsnextpages}%
80   }%
81 }

```

`\@gls@loadlong`

```

82 \newcommand*\@gls@loadlong{\RequirePackage{glossary-long}}

```

`nolong` This option prevents from being loaded. This means that the glossary styles that use the longtable environment will not be available. This option is provided to reduce overhead caused by loading unrequired packages.

```

83 \@gls@declareoption{nolong}{\renewcommand*\@gls@loadlong{}}

```

`\@gls@loadsuper`

The package isn't loaded if isn't installed.

```

84 \IfFileExists{supertabular.sty}{%
85   \newcommand*\@gls@loadsuper{\RequirePackage{glossary-super}}}{%
86   \newcommand*\@gls@loadsuper{}}

```

`nosuper` This option prevents from being loaded. This means that the glossary styles that use the supertabular environment will not be available. This option is provided to reduce overhead caused by loading unrequired packages.

```

87 \@gls@declareoption{nosuper}{\renewcommand*\@gls@loadsuper{}}

```

`\@gls@loadlist`

```

88 \newcommand*\@gls@loadlist{\RequirePackage{glossary-list}}

```

`nolist` This option prevents from being loaded (to reduce overheads if required). Naturally, the styles defined in will not be available if this option is used.

```

89 \@gls@declareoption{nolist}{\renewcommand*\@gls@loadlist{}}

```

`\@gls@loadtree`

```

90 \newcommand*\@gls@loadtree{\RequirePackage{glossary-tree}}

```


notree This option prevents from being loaded (to reduce overheads if required). Naturally, the styles defined in will not be available if this option is used.

```
91 \@gls@declareoption{notree}{\renewcommand*{\@gls@loadtree}{}}
```

nostyles Provide an option to suppress all the predefined styles (in the event that the user has custom styles that are not dependent on the predefined styles).

```
92 \@gls@declareoption{nostyles}{%
93   \renewcommand*{\@gls@loadlong}{}%
94   \renewcommand*{\@gls@loadsuper}{}%
95   \renewcommand*{\@gls@loadlist}{}%
96   \renewcommand*{\@gls@loadtree}{}%
97   \let\@glossary@default@style\relax
98 }
```

\glspostdescription The description terminator is given by `\glspostdescription` (except for the 3 and 4 column styles). This is a full stop by default. The spacefactor is adjusted in case the description ends with an upper case letter. (Patch provided by Michael Pock.)

```
99 \newcommand*{\glspostdescription}{%
100   \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi
101 }
```

nopostdot Boolean option to suppress post description dot

```
102 \define@boolkey{glossaries.sty}[gls]{nopostdot}[true]{}
103 \glsnopostdotfalse
```

nogroupskip Boolean option to suppress vertical space between groups in the pre-defined styles.

```
104 \define@boolkey{glossaries.sty}[gls]{nogroupskip}[true]{}
105 \glsnogroupskipfalse
```

ucmark Boolean option to determine whether or not to use upper case in definition of `\gls glossarymark`

```
106 \define@boolkey{glossaries.sty}[gls]{ucmark}[true]{}
107 \@ifclassloaded{memoir}
108 {%
109   \glsucmarktrue
110 }%
111 {%
112   \glsucmarkfalse
113 }
```

entrycounter Defines a counter that can be used in the standard glossary styles to number each (main) entry. If true, this will define a counter called `glossaryentry`.

```
114 \define@boolkey{glossaries.sty}[gls]{entrycounter}[true]{}
115 \glsentrycounterfalse
```

entrycounterwithin This option can be used to set a parent counter for glossaryentry. This option automatically sets entrycounter=true.

```
116 \define@key{glossaries.sty}{counterwithin}{%
117   \renewcommand*{\@gls@counterwithin}{#1}%
118   \glsentrycountertrue
119 }
```

\@gls@counterwithin The default value is no parent counter:

```
120 \newcommand*{\@gls@counterwithin}{}
```

subentrycounter Define a counter that can be used in the standard glossary styles to number each level 1 entry. If true, this will define a counter called glossarysubentry.

```
121 \define@boolkey{glossaries.sty}[gls]{subentrycounter}[true]{%
122 \glssubentrycounterfalse
```

\@glo@default@sorttype Initialise default sort for \printnoidxglossary

```
123 \newcommand*{\@glo@default@sorttype}{standard}
```

sort Define the sort method: sort=standard (default), sort=def (order of definition) or sort=use (order of use).

```
124 \define@choicekey{glossaries.sty}{sort}{standard,def,use}{%
125   \renewcommand*{\@glo@default@sorttype}{#1}%
126   \csname @gls@setupsort@#1\endcsname
127 }
```

\@gls@prestandardsort `\@gls@prestandardsort{<sort cs>}{<type>}{<label>}`

Allow user to hook into sort mechanism. The first argument *<sort cs>* is the temporary control sequence containing the sort value before it has been sanitized and had `makeindex/xindy` special characters escaped.

```
128 \newcommand*{\@gls@prestandardsort}[3]{%
129   \glsdosanitizesort
130 }
```

@setupsort@standard Set up the macros for default sorting.

```
131 \newcommand*{\@gls@setupsort@standard}{%
```

Store entry information when it's defined.

```
132   \def\do@glo@storeentry{\@glo@storeentry}%
```

No count register required for standard sort.

```
133   \def\@gls@defsortcount##1{%
```

Sort according to sort key (\@glo@sort) if provided otherwise sort according to the entry's name (\@glo@name). (First argument glossary type, second argument entry label.)

```
134   \def\@gls@defsort##1##2{%
```

```

135 \ifx\@glo@sort\@glsdefaultsort
136 \let\@glo@sort\@glo@name
137 \fi

138 \let\glsdosanitizesort\@gls@sanitizesort
139 \glsprestandardsort{\@glo@sort}{##1}{##2}%
140 \expandafter\protected@xdef\csname glo@##2@sort\endcsname{\@glo@sort}%
141 }%

```

Don't need to do anything when the entry is used.

```

142 \def\@gls@setsort##1{%
143 }

```

Set standard sort as the default:

```

144 \@gls@setupsort@standard

```

`\glsnumberfmt` Format the number used as the sort key by `sort=def` and `sort=use`. Defaults to six digit numbering.

```

145 \newcommand*\glsnumberfmt[1]{%
146 \ifnum#1<100000 0\fi
147 \ifnum#1<10000 0\fi
148 \ifnum#1<1000 0\fi
149 \ifnum#1<100 0\fi
150 \ifnum#1<10 0\fi
151 \number#1%
152 }

```

`\@gls@setupsort@def` Set up the macros for order of definition sorting.

```

153 \newcommand*\@gls@setupsort@def{%

```

Store entry information when it's defined.

```

154 \def\do@glo@storeentry{\@glo@storeentry}%

```

Defined count register associated with the glossary.

```

155 \def\@gls@defsortcount##1{%
156 \expandafter\global
157 \expandafter\newcount\csname glossary@##1@sortcount\endcsname
158 }%

```

Increment count register associated with the glossary and use as the sort key.

```

159 \def\@gls@defsort##1##2{%
160 \expandafter\global\expandafter
161 \advance\csname glossary@##1@sortcount\endcsname by 1\relax
162 \expandafter\protected@xdef\csname glo@##2@sort\endcsname{%
163 \expandafter\glsnumberfmt
164 {\csname glossary@##1@sortcount\endcsname}}%
165 }%

```

Don't need to do anything when the entry is used.

```

166 \def\@gls@setsort##1{%
167 }

```

```

\@gls@setupsort@use  Set up the macros for order of use sorting.
168 \newcommand*{\@gls@setupsort@use}{%
    Don't store entry information when it's defined.
169 \let\do@glo@storeentry\@gobble
    Defined count register associated with the glossary.
170 \def\@gls@defsortcount##1{%
171 \expandafter\global
172 \expandafter\newcount\csname glossary@##1@sortcount\endcsname
173 }%
    Initialise the sort key to empty.
174 \def\@gls@defsort##1##2{%
175 \expandafter\gdef\csname glo@##2@sort\endcsname{}}%
176 }%
    If the sort key hasn't been set, increment the counter associated with the glossary and set the sort key.
177 \def\@gls@setsort##1{%
    Get the parent, if one exists
178 \edef\@glo@parent{\csname glo@##1@parent\endcsname}%
    Set the information for the parent entry if not already done.
179 \ifx\@glo@parent\@empty
180 \else
181 \expandafter\@gls@setsort\expandafter{\@glo@parent}%
182 \fi
    Set index information for this entry
183 \edef\@glo@type{\csname glo@##1@type\endcsname}%
184 \edef\@gls@tmp{\csname glo@##1@sort\endcsname}%
185 \ifx\@gls@tmp\@empty
186 \expandafter\global\expandafter
187 \advance\csname glossary@\@glo@type @sortcount\endcsname by 1\relax
188 \expandafter\protected@xdef\csname glo@##1@sort\endcsname{%
189 \expandafter\glssortnumberfmt
190 {\csname glossary@\@glo@type @sortcount\endcsname}}%
191 \@glo@storeentry{##1}%
192 \fi
193 }%
194 }

```

`\glsdefmain` Define the main glossary. This will be the first glossary to be displayed when using `\printglossaries`. The default extensions conflict if used with `doc`, so provide different extensions if `doc` loaded. (If these extensions are inappropriate, use `nomain` and manually define the main glossary with the desired extensions.)

```

195 \newcommand*{\glsdefmain}{%
196 \if@gls@docloaded

```

```

197   \newglossary[glg2]{main}{gls2}{glo2}{\glossaryname}%
198   \else
199   \newglossary{main}{gls}{glo}{\glossaryname}%
200   \fi

Define hook to set the toc title when translator is in use.
201   \newcommand*{\gls@tr@set@main@toctitle}{%
202     \translatelet{\glossarytoctitle}{Glossary}%
203   }%
204 }

```

Keep track of the default glossary. This is initialised to the main glossary, but can be changed if for some reason you want to make a secondary glossary the main glossary. This affects any commands that can optionally take a glossary name as an argument (or as the value of the type key in a key-value list). This was mainly done so that `\loadglsentries` can temporarily change `\glsdefaulttype` while it loads a file containing new glossary entries (see [subsection 1.9](#)).

`\glsdefaulttype`

```
205 \newcommand*{\glsdefaulttype}{main}
```

Keep track of which glossary the acronyms are in. This is initialised to `\glsdefaulttype`, but is changed by the acronym package option.

`\acronymtype`

```
206 \newcommand*{\acronymtype}{\glsdefaulttype}
```

`nomain` The `nomain` option suppress the creation of the main glossary.

```

207 \@gls@declareoption{nomain}{%
208   \let\glsdefaulttype\relax
209   \renewcommand*{\glsdefmain}{}%
210 }

```

`acronym` The `acronym` option sets an associated conditional which is used in [subsection 1.16](#) to determine whether or not to define a separate glossary for acronyms.

```

211 \define@boolkey{glossaries.sty}[gls]{acronym}[true]{%
212   \ifglsacronym
213     \renewcommand{\@gls@do@acronymsdef}{%
214       \DeclareAcronymList{acronym}%
215       \newglossary[alg]{acronym}{acr}{acn}{\acronymname}%
216       \renewcommand*{\acronymtype}{acronym}%

```

Define hook to set the toc title when translator is in use.

```

217   \newcommand*{\gls@tr@set@acronym@toctitle}{%
218     \translatelet{\glossarytoctitle}{Acronyms}%
219   }%
220 }%

```

```

221 \else
222 \let\@gls@do@acronymsdef\relax
223 \fi
224 }

```

`\printacronyms` Define `\printacronyms` at the start of the document if acronym is set and compatibility mode isn't on and `\printacronyms` hasn't already been defined.

```

225 \AtBeginDocument{%
226 \ifglsacronym
227 \ifbool{glscompatible-3.07}%
228 {}%
229 {%
230 \providecommand*\printacronyms[1][{}]{%
231 \printglossary[type=\acronymtype,#1]}%
232 }%
233 \fi
234 }

```

`@gls@do@acronymsdef` Set default value

```

235 \newcommand*\@gls@do@acronymsdef{}

```

`acronyms` Provide a synonym for `acronym=true` that can be passed via the document class options.

```

236 \@gls@declareoption{acronyms}{%
237 \glsacronymtrue
238 \renewcommand*\@gls@do@acronymsdef{%
239 \DeclareAcronymList{acronym}%
240 \newglossary[alg]{acronym}{acr}{acn}{\acronymname}%
241 \renewcommand*\acronymtype{acronym}%

```

Define hook to set the toc title when translator is in use.

```

242 \newcommand*\gls@tr@set@acronym@toctitle{%
243 \translatelet{\glossarytoctitle}{Acronyms}%
244 }%
245 }%
246 }

```

`\@glsacronymlists` Comma-separated list of glossary labels indicating which glossaries contain acronyms. Note that `\SetAcronymStyle` must be used after adding labels to this macro.

```

247 \newcommand*\@glsacronymlists{}

```

`\@addtoacronymlists`

```

248 \newcommand*\@addtoacronymlists[1]{%
249 \ifx\@glsacronymlists\@empty
250 \protected@xdef\@glsacronymlists{#1}%
251 \else
252 \protected@xdef\@glsacronymlists{\@glsacronymlists,#1}%

```

```
253 \fi
254 }
```

`\DeclareAcronymList` Identifies the named glossary as a list of acronyms and adds to the list. (Doesn't check if the glossary exists, but checks if label already in list. Use `\SetAcronymStyle` after identifying all the acronym lists.)

```
255 \newcommand*\DeclareAcronymList}[1]{%
256 \glsIfListOfAcronyms{#1}{\@addtoacronymlists{#1}}%
257 }
```

`\glsIfListOfAcronyms` `\glsIfListOfAcronyms{<label>}{<true part>}{<false part>}`

Determines if the glossary with the given label has been identified as being a list of acronyms.

```
258 \newcommand\glsIfListOfAcronyms}[1]{%
259 \edef\@do@gls@islistofacronyms{%
260 \noexpand\@gls@islistofacronyms{#1}{\@glsacronymlists}}%
261 \@do@gls@islistofacronyms
262 }
```

Internal command requires label and list to be expanded:

```
263 \newcommand\@gls@islistofacronyms}[4]{%
264 \def\gls@islistofacronyms##1,#1,##2\end@gls@islistofacronyms{%
265 \def\@before{##1}\def\@after{##2}}%
266 \gls@islistofacronyms,#2,#1,\@nil\end@gls@islistofacronyms
267 \ifx\@after\@nnil
```

Not found

```
268 #4%
269 \else
```

Found

```
270 #3%
271 \fi
272 }
```

`\if@glsisacronymlist` Convenient boolean.

```
273 \newif\if@glsisacronymlist
```

`@checkisacronymlist` Sets the above boolean if argument is a label representing a list of acronyms.

```
274 \newcommand*\checkisacronymlist}[1]{%
275 \glsIfListOfAcronyms{#1}%
276 {\@glsisacronymlisttrue}{\@glsisacronymlistfalse}%
277 }
```

`\SetAcronymLists` Sets the “list of acronyms” list. Argument must be a comma-separated list of glossary labels. (Doesn't check at this point if the glossaries exists.)

```
278 \newcommand*\SetAcronymLists}[1]{%
```

```

279 \renewcommand*{\@glsacronymlists}{#1}%
280 }

```

acronymlists

```

281 \define@key{glossaries.sty}{acronymlists}{%
282 \DeclareAcronymList{#1}%
283 }

```

The default counter associated with the numbers in the glossary is stored in `\glscounter`. This is initialised to the page counter. This is used as the default counter when a new glossary is defined, unless a different counter is specified in the optional argument to `\newglossary` (see [subsection 1.6](#)).

\glscounter

```

284 \newcommand{\glscounter}{page}

```

counter The counter option changes the default counter. (This just redefines `\glscounter`.)

```

285 \define@key{glossaries.sty}{counter}{%
286 \renewcommand*{\glscounter}{#1}%
287 }

```

\@gls@nohyperlist

```

288 \newcommand*{\@gls@nohyperlist}{}

```

\GlsDeclareNoHyperList

```

289 \newcommand*{\GlsDeclareNoHyperList}[1]{%
290 \ifdefempty\@gls@nohyperlist
291 {%
292 \renewcommand*{\@gls@nohyperlist}{#1}%
293 }%
294 {%
295 \appto\@gls@nohyperlist{,#1}%
296 }%
297 }

```

nohypertypes

```

298 \define@key{glossaries.sty}{nohypertypes}{%
299 \GlsDeclareNoHyperList{#1}%
300 }

```

\GlossariesWarning

Prints a warning message.

```

301 \newcommand*{\GlossariesWarning}[1]{%
302 \PackageWarning{glossaries}{#1}%
303 }

```

\GlossariesWarningNoLine

Prints a warning message without the line number.

```

304 \newcommand*{\GlossariesWarningNoLine}[1]{%
305 \PackageWarningNoLine{glossaries}{#1}%
306 }

```


nowarn Define package option to suppress warnings

```

307 \@gls@declareoption{nowarn}{%
308   \renewcommand*{\GlossariesWarning}[1]{}%
309   \renewcommand*{\GlossariesWarningNoLine}[1]{}%
310 }
```

@warnonglossdefined Issue a warning if overriding \printglossary

```

311 \newcommand*{\@gls@warnonglossdefined}{%
312   \GlossariesWarning{Overriding \string\printglossary}%
313 }
```

warnontheGLOSSdefined Issue a warning if overriding theGLOSSary

```

314 \newcommand*{\@gls@warnontheGLOSSdefined}{%
315   \GlossariesWarning{Overriding 'theGLOSSary' environment}%
316 }
```

noredefwarn Suppress warning on redefinition of \printglossary

```

317 \@gls@declareoption{noredefwarn}{%
318   \renewcommand*{\@gls@warnonglossdefined}{}%
319   \renewcommand*{\@gls@warnontheGLOSSdefined}{}%
320 }
```

As from version 3.08a, the only information written to the external glossary files are the label and sort values. Therefore, now, the only sanitize option that makes sense is the one for the sort key. so the sanitize option is now deprecated and there is only a sanitizesort option.

\@gls@sanitizedesc

```

321 \newcommand*{\@gls@sanitizedesc}{%
322 }
```

\glssetexpandfield `\glssetexpandfield{<field>}`

Sets field to always expand.

```

323 \newcommand*{\glssetexpandfield}[1]{%
324   \csdef{gls@assign@#1@field}##1##2{%
325     \@gls@expand@field{##1}{#1}{##2}%
326   }%
327 }
```

\glssetnoexpandfield `\glssetnoexpandfield{<field>}`

Sets field to never expand.

```

328 \newcommand*{\glssetnoexpandfield}[1]{%
329   \csdef{gls@assign@#1@field}##1##2{%
```

```

330   \@@gls@noexpand@field{##1}{#1}{##2}%
331   }%
332 }

s@assign@type@field  The type must always be expandable.
333 \glssetexpandfield{type}

s@assign@desc@field  The description is not expanded by default:
334 \glssetnoexpandfield{desc}

gn@descplural@field
335 \glssetnoexpandfield{descplural}

\@gls@sanitizename
336 \newcommand*{\@gls@sanitizename}{}

s@assign@name@field  Don't expand name by default.
337 \glssetnoexpandfield{name}

@gls@sanitizesymbol
338 \newcommand*{\@gls@sanitizesymbol}{}

assign@symbol@field  Don't expand symbol by default.
339 \glssetnoexpandfield{symbol}

@symbolplural@field
340 \glssetnoexpandfield{symbolplural}

Sanitizing stuff:

\@gls@sanitizesort
341 \newcommand*{\@gls@sanitizesort}{%
342   \ifglssanitizesort
343     \@gls@sanitizesort
344   \else
345     \@gls@nosanitizesort
346   \fi
347 }

\@@gls@sanitizesort
348 \newcommand*\@@gls@sanitizesort{%
349   \@onelevel@sanitize\@glo@sort
350 }

@gls@nosanitizesort
351 \newcommand*{\@gls@nosanitizesort}{}

```

@noidx@sanitizesort Remove braces around first character (if present) before sanitizing.

```
352 \newcommand*\@gls@noidx@sanitizesort{%
353   \ifdefvoid\@glo@sort
354   {}%
355   {%
356     \expandafter\@@gls@noidx@sanitizesort\@glo@sort\gls@end@sanitizesort
357   }%
358 }
359 \def\@@gls@noidx@sanitizesort#1#2\gls@end@sanitizesort{%
360   \def\@glo@sort{#1#2}%
361   \@onelevel@sanitize\@glo@sort
362 }
```

noidx@nosanitizesort

```
363 \newcommand*\@@gls@noidx@nosanitizesort{%
364   \ifdefvoid\@glo@sort
365   {}%
366   {%
367     \expandafter\@@gls@noidx@no@sanitizesort\@glo@sort\gls@end@sanitizesort
368   }%
369 }
370 \def\@@gls@noidx@no@sanitizesort#1#2\gls@end@sanitizesort{%
371   \bgroup
372     \glsnoidxstripaccents
373     \protected@xdef\@@glo@sort{#1#2}%
374   \egroup
375   \let\@glo@sort\@@glo@sort
376 }
```

lsnoidxstripaccents

```
377 \newcommand*\glsnoidxstripaccents{%
378   \let\IeC\@firstofone
379   \let\'@\@firstofone
380   \let\'\@firstofone
381   \let\~@\@firstofone
382   \let\"@\@firstofone
383   \let\u@\@firstofone
384   \let\t@\@firstofone
385   \let\d@\@firstofone
386   \let\r@\@firstofone
387   \let\=\@firstofone
388   \let\.\@firstofone
389   \let\~@\@firstofone
390   \let\v@\@firstofone
391   \let\H@\@firstofone
392   \let\c@\@firstofone
393   \let\b@\@firstofone
394   \def\AE{AE}%
395   \def\ae{ae}%
```

```

396 \def\OE{OE}%
397 \def\oe{oe}%
398 \def\AA{AA}%
399 \def\aa{aa}%
400 \def\L{L}%
401 \def\l{l}%
402 \def\O{O}%
403 \def\o{o}%
404 \def\SS{SS}%
405 \def\ss{ss}%
406 \def\th{th}%
407 }

```

Before defining the `sanitize` package option, The key-value list for the `sanitize` value needs to be defined. These are all boolean keys. If they are not given a value, assume true.

```

408 \define@boolkey[glS]{sanitize}{description}[true]{%
409   \GlossariesWarning{sanitize={description} package option deprecated}%
410   \ifglS@sanitize@description
411     \glSsetnoexpandfield{desc}%
412     \glSsetnoexpandfield{descplural}%
413   \else
414     \glSsetexpandfield{desc}%
415     \glSsetexpandfield{descplural}%
416   \fi
417 }

418 \define@boolkey[glS]{sanitize}{name}[true]{%
419   \GlossariesWarning{sanitize={name} package option deprecated}%
420   \ifglS@sanitize@name
421     \glSsetnoexpandfield{name}%
422   \else
423     \glSsetexpandfield{name}%
424   \fi
425 }

426 \define@boolkey[glS]{sanitize}{symbol}[true]{%
427   \GlossariesWarning{sanitize={symbol} package option deprecated}%
428   \ifglS@sanitize@symbol
429     \glSsetnoexpandfield{symbol}%
430     \glSsetnoexpandfield{symbolplural}%
431   \else
432     \glSsetexpandfield{symbol}%
433     \glSsetexpandfield{symbolplural}%
434   \fi
435 }

```

`sanitizesort`

```

436 \define@boolkey{glossaries.sty}[glS]{sanitizesort}[true]{%
437   \ifglSsanitizesort

```

```

438 \glsssetnoexpandfield{sortvalue}%
439 \renewcommand*{\@gls@noidx@setsanitizesort}{%
440 \glssanitizesorttrue
441 \glsssetnoexpandfield{sortvalue}%
442 }%
443 \else
444 \glsssetexpandfield{sortvalue}%
445 \renewcommand*{\@gls@noidx@setsanitizesort}{%
446 \glssanitizesortfalse
447 \glsssetexpandfield{sortvalue}%
448 }%
449 \fi
450 }

```

Default setting:

```

451 \glssanitizesorttrue
452 \glsssetnoexpandfield{sortvalue}%

```

`\@gls@setsanitizesort` Default behaviour for `\makenoidxglossaries` is `sanitizesort=false`.

```

453 \newcommand*{\@gls@noidx@setsanitizesort}{%
454 \glssanitizesortfalse
455 \glsssetexpandfield{sortvalue}%
456 }

457 \define@choicekey[gls]{sanitize}{sort}{true,false}[true]{%
458 \setbool{glssanitizesort}{#1}%
459 \ifglssanitizesort
460 \glsssetnoexpandfield{sortvalue}%
461 \else
462 \glsssetexpandfield{sortvalue}%
463 \fi
464 \GlossariesWarning{sanitize={sort} package option
465 deprecated. Use sanitizesort instead}%
466 }

```

`sanitize`

```

467 \define@key{glossaries.sty}{sanitize}[description=true,symbol=true,name=true]{%
468 \ifthenelse{\equal{#1}{none}}{%
469 {%
470 \GlossariesWarning{sanitize package option deprecated}%
471 \glsssetexpandfield{name}%
472 \glsssetexpandfield{symbol}%
473 \glsssetexpandfield{symbolplural}%
474 \glsssetexpandfield{desc}%
475 \glsssetexpandfield{descplural}%
476 }%
477 }%
478 \setkeys[gls]{sanitize}{#1}%
479 }%
480 }

```

`\ifglstranslate` As from version 3.13a, the translator package option is a choice rather than boolean option so now need to define conditional:

```
481 \newif\ifglstranslate
```

`ls@notranslatorhook`

```
482 \newcommand*\@gls@notranslatorhook{}
```

`notranslate` Provide a synonym for `translate=false` that can be passed via the document class.

```
483 \@gls@declareoption{notranslate}{%
484   \glstranslatefalse
485   \let\@gls@notranslatorhook\relax
486 }
```

`translate` Define `translate` option. If false don't set up multi-lingual support.

```
487 \define@choicekey{glossaries.sty}{translate}[\val\nr]%
488 {true,false,babel}[true]%
489 {%
490   \ifcase\nr\relax
491     \glstranslatetrue
492   \or
493     \glstranslatefalse
494     \let\@gls@notranslatorhook\relax
495   \or
496     \glstranslatefalse
497     \def\@gls@notranslatorhook{\RequirePackage{glossaries-babel}}%
498   \fi
499 }
```

Set the default value:

```
500 \glstranslatefalse
501 \@ifpackageloaded{translator}%
502   {\glstranslatetrue}%
503   {%
504     \@ifpackageloaded{polyglossia}%
505       {\glstranslatetrue}%
506       {%
507         \@ifpackageloaded{babel}{\glstranslatetrue}{}}%
508       }%
509 }
```

`indexonlyfirst` Set whether to only index on first use.

```
510 \define@boolkey{glossaries.sty}[gls]{indexonlyfirst}[true]{}
511 \glsindexonlyfirstfalse
```

`hyperfirst` Set whether or not terms should have a hyperlink on first use.

```
512 \define@boolkey{glossaries.sty}[gls]{hyperfirst}[true]{}
513 \glshyperfirsttrue
```

`\@gls@setacrstyle` Keep track of whether an acronym style has been set (for the benefit of `\setupglossaries`):

```
514 \newcommand*\@gls@setacrstyle{}
```

`footnote` Set the long form of the acronym in footnote on first use.

```
515 \define@boolkey{glossaries.sty}[glsacr]{footnote}[true]{%
516   \ifbool{glsacrdescription}%
517   {}%
518   {%
519     \renewcommand*\@gls@sanitizedesc}{}%
520   }%
521   \renewcommand*\@gls@setacrstyle{\SetAcronymStyle}%
522 }
```

`description` Allow acronyms to have a description (needs to be set using the description key in the optional argument of `\newacronym`).

```
523 \define@boolkey{glossaries.sty}[glsacr]{description}[true]{%
524   \renewcommand*\@gls@sanitizesymbol}{}%
525   \renewcommand*\@gls@setacrstyle{\SetAcronymStyle}%
526 }
```

`smallcaps` Define `\newacronym` to set the short form in small capitals.

```
527 \define@boolkey{glossaries.sty}[glsacr]{smallcaps}[true]{%
528   \renewcommand*\@gls@sanitizesymbol}{}%
529   \renewcommand*\@gls@setacrstyle{\SetAcronymStyle}%
530 }
```

`smaller` Define `\newacronym` to set the short form using `\smaller` which obviously needs to be defined by loading the appropriate package.

```
531 \define@boolkey{glossaries.sty}[glsacr]{smaller}[true]{%
532   \renewcommand*\@gls@sanitizesymbol}{}%
533   \renewcommand*\@gls@setacrstyle{\SetAcronymStyle}%
534 }
```

`dua` Define `\newacronym` to always use the long forms (i.e. don't use acronyms)

```
535 \define@boolkey{glossaries.sty}[glsacr]{dua}[true]{%
536   \renewcommand*\@gls@sanitizesymbol}{}%
537   \renewcommand*\@gls@setacrstyle{\SetAcronymStyle}%
538 }
```

`shortcuts` Define acronym shortcuts.

```
539 \define@boolkey{glossaries.sty}[glsacr]{shortcuts}[true]{}
```

`\glsorder` Stores the glossary ordering. This may either be “word” or “letter”. This passes the relevant information to `makeglossaries`. The default is word ordering.

```
540 \newcommand*\glsorder{word}
```

`\@glsorder` The ordering information is written to the auxiliary file for `makeglossaries`, so ignore the auxiliary information.

```
541 \newcommand*{\@glsorder}[1]{}
```

order

```
542 \define@choicekey{glossaries.sty}{order}{word,letter}{%}
```

```
543 \def\glsorder{#1}}
```

`\ifglsxindy` Provide boolean to determine whether `xindy` or `makeindex` will be used to sort the glossaries.

```
544 \newif\ifglsxindy
```

The default is `makeindex`:

```
545 \glsxindyfalse
```

`makeindex` Define package option to specify that `makeindex` will be used to sort the glossaries:

```
546 \@gls@declareoption{makeindex}{\glsxindyfalse}
```

The `xindy` package option may have a value which in turn can be a key=value list. First define the keys for this sub-list. The boolean `glsnumbers` determines whether to automatically add the `glsnumbers` letter group.

```
547 \define@boolkey[gls]{xindy}{glsnumbers}[true]{}
```

```
548 \gls@xindy@glsnumberstrue
```

`\@xdy@main@language` Define what language to use for each glossary type (if a language is not defined for a particular glossary type the language specified for the main glossary is used.)

```
549 \def\@xdy@main@language{\language}%
```

Define key to set the language

```
550 \define@key[gls]{xindy}{language}{\def\@xdy@main@language{#1}}
```

`\gls@codepage` Define the code page. If `\inputencodingname` is defined use that, otherwise have initialise with no codepage.

```
551 \ifcsundef{inputencodingname}{%
```

```
552 \def\gls@codepage{}{}}%
```

```
553 \def\gls@codepage{\inputencodingname}
```

```
554 }
```

Define a key to set the code page.

```
555 \define@key[gls]{xindy}{codepage}{\def\gls@codepage{#1}}
```

`xindy` Define package option to specify that `xindy` will be used to sort the glossaries:

```
556 \define@key{glossaries.sty}{xindy}[]{}%
```

```
557 \glsxindytrue
```

```
558 \setkeys[gls]{xindy}{#1}%
```

```
559 }
```


`xindygloss` Provide a synonym for `xindy` that can be passed via the document class options.

```
560 \@gls@declareoption{xindygloss}{%
561   \glsxindytrue
562 }
```

`xindynoglsnumbers` Provide a synonym for `xindy=glsnumbers=false` that can be passed via the document class options.

```
563 \@gls@declareoption{xindynoglsnumbers}{%
564   \glsxindytrue
565   \gls@xindy@glsnumbersfalse
566 }
```

`automake` If this setting is on, automatically run `makeindex/xindy` at the end of the document. Must be used with `\makeglossaries`. Default is false.

```
567 \define@boolkey{glossaries.sty}[gls]{automake}[true]{%
568   \ifglsautomake
569     \renewcommand*{\@gls@doautomake}{%
570       \PackageError{glossaries}{You must use
571         \string\makeglossaries\space with automake=true}
572       {%
573         Either remove the automake=true setting or
574         add \string\makeglossaries\space to your document preamble.%
575       }%
576     }%
577   \else
578     \renewcommand*{\@gls@doautomake}{}%
579   \fi
580 }
581 \glsautomakefalse
```

`\@gls@doautomake`

```
582 \newcommand*{\@gls@doautomake}{%
583 \AtEndDocument{\@gls@doautomake}
```

`savewrites` The `savewrites` package option is provided to save on the number of write registers.

```
584 \define@boolkey{glossaries.sty}[gls]{savewrites}[true]{%
585   \ifglssavewrites
586     \renewcommand*{\glswritefiles}{\@glswritefiles}%
587   \else
588     \let\glswritefiles\@empty
589   \fi
590 }
```

Set default:

```
591 \glssavewritesfalse
592 \let\glswritefiles\@empty
```

compatible-3.07

```
593 \define@boolkey{glossaries.sty}[gls]{compatible-3.07}[true]{%
594 \boolfalse{glscompatible-3.07}}
```

compatible-2.07

```
595 \define@boolkey{glossaries.sty}[gls]{compatible-2.07}[true]{%
  Also set 3.07 compatibility if this option is set.
596 \ifbool{glscompatible-2.07}%
597 {%
598 \booltrue{glscompatible-3.07}%
599 }%
600 {}%
601 }
602 \boolfalse{glscompatible-2.07}
```

symbols Create a “symbols” glossary type

```
603 \@gls@declareoption{symbols}{%
604 \let\@gls@do@symbolsdef\@gls@symbolsdef
605 }
```

Default is not to define the symbols glossary:

```
606 \newcommand*\@gls@do@symbolsdef{}
```

\@gls@symbolsdef

```
607 \newcommand*\@gls@symbolsdef{%
608 \newglossary[slg]{symbols}{sls}{slo}{\glssymbolsgroupname}%
609 \newcommand*\@printsymbols[1][\printglossary[type=symbols,##1]]%
```

Define hook to set the toc title when translator is in use.

```
610 \newcommand*\@gls@tr@set@symbols@toctitle{%
611 \translatelet{\glossarytoctitle}{Symbols (glossaries)}%
612 }%
613 }%
```

numbers Create a “symbols” glossary type

```
614 \@gls@declareoption{numbers}{%
615 \let\@gls@do@numbersdef\@gls@numbersdef
616 }
```

Default is not to define the numbers glossary:

```
617 \newcommand*\@gls@do@numbersdef{}
```

\@gls@numbersdef

```
618 \newcommand*\@gls@numbersdef{%
619 \newglossary[nlg]{numbers}{nls}{nlo}{\glsnumbersgroupname}%
620 \newcommand*\@printnumbers[1][\printglossary[type=numbers,##1]]%
```

Define hook to set the toc title when translator is in use.

```
621 \newcommand*{\gls@tr@set@numbers@toctitle}{%
622   \translatelet{\glossarytoctitle}{Numbers (glossaries)}%
623 }%
624 }%
```

index Create an “index” glossary type

```
625 \@gls@declareoption{index}{%
626   \let\@gls@do@indexdef\@gls@indexdef
627 }
```

Default is not to define index glossary:

```
628 \newcommand*{\@gls@do@indexdef}{}
```

\@gls@indexdef \indexname isn't set by glossaries.

```
629 \newcommand*{\@gls@indexdef}{%
630   \newglossary[ilg]{index}{ind}{idx}{\indexname}%
631   \newcommand*{\printindex}[1] [] {\printglossary[type=index,##1]}%
632   \newcommand*{\newterm}[2] [] {%
633     \newglossaryentry{##2}%
634     {type={index},name={##2},description={\nopostdesc},##1}}
635 }%
```

Process package options. First process any options that have been passed via the document class.

```
636 \@for\CurrentOption :=\@declaredoptions\do{%
637   \ifx\CurrentOption\@empty
638     \else
639       \@expandtwoargs
640       \in@ {,\CurrentOption ,}{,\@classoptionslist,\@curroptions,}%
641       \ifin@
642         \@use@ption
643         \expandafter \let\csname ds@\CurrentOption\endcsname\@empty
644       \fi
645     \fi
646 }
```

Now process options passed to the package:

```
647 \ProcessOptionsX
```

Load backward compatibility stuff:

```
648 \RequirePackage{glossaries-compatible-307}
```

\setupglossaries Provide way to set options after package has been loaded. However, some options must be set before \ProcessOptionsX, so they have to be disabled:

```
649 \disable@keys{glossaries.sty}{compatible-2.07,%
650 xindy,xindygloss,xindynoglsnumbers,makeindex,%
651 acronym,translate,notranslate,nolong,nosuper,notree,nostyles,nomain}
```

Now define `\setupglossaries`:

```
652 \newcommand*\setupglossaries}[1]{%
653   \renewcommand*\@gls@setacrstyle}{}%
654   \ifglsacrshortcuts
655     \def\@gls@setupshortcuts{\glsacrshortcutstrue}%
656   \else
657     \def\@gls@setupshortcuts{%
658       \ifglsacrshortcuts
659         \DefineAcronymSynonyms
660       \fi
661     }%
662   \fi
663   \glsacrshortcutsfalse
664   \let\@gls@do@numbersdef\relax
665   \let\@gls@do@symbolssdef\relax
666   \let\@gls@do@indexdef\relax
667   \let\@gls@do@acronymsdef\relax
668   \setkeys{glossaries.sty}{#1}%
669   \@gls@setacrstyle
670   \@gls@setupshortcuts
671   \@gls@do@acronymsdef
672   \@gls@do@numbersdef
673   \@gls@do@symbolssdef
674   \@gls@do@indexdef
675 }
```

If package is loaded, check to see if is installed, but only if translation is required.

```
676 \ifglstranslate
677   \@ifpackageloaded{polyglossia}%
678   {%
679     }%
680   {%
681     \@ifpackageloaded{babel}%
682     {%
683       \IfFileExists{translator.sty}%
684       {%
685         \RequirePackage{translator}%
686       }%
687     }%
688   }%
689   {}
690 }
691 \fi
```

If chapters are defined and the user has requested the section counter as a package option, `\@chapter` will be modified so that it adds a section. $\langle n \rangle . 0$

target, otherwise entries placed before the first section of a chapter will have undefined links.

The same problem will also occur if a lower sectional unit is used, but this is less likely to happen. If it does, or if you change `\glscounter` to `section` later, you will have to specify a different counter for the entries that give rise to a name- $\langle section-level \rangle . \langle n \rangle . 0$ non-existent warning (e.g. `\gls[counter=chapter]{label}`).

```

692 \ifthenelse{\equal{\glscounter}{section}}{%
693 {%
694   \ifcsundef{chapter}{}%
695   {%
696     \let\@gls@old@chapter\@chapter
697     \def\@chapter[#1]#2{\@gls@old@chapter[#1]{#2}%
698     \ifcsundef{hyperdef}{}{\hyperdef{section}{\thesection}}}%
699   }%
700 }%
701 {}

```

`\@gls@onlypremakeg` Some commands only have an effect when used before `\makeglossaries`. So define a list of commands that should be disabled after `\makeglossaries`

```

702 \newcommand*\@gls@onlypremakeg{}

```

`\@onlypremakeg` Adds the specified control sequence to the list of commands that must be disabled after `\makeglossaries`.

```

703 \newcommand*\@onlypremakeg[1]{%
704   \ifx\@gls@onlypremakeg\@empty
705     \def\@gls@onlypremakeg{#1}%
706   \else
707     \expandafter\toks@\expandafter{\@gls@onlypremakeg}%
708     \edef\@gls@onlypremakeg{\the\toks@,\noexpand#1}%
709   \fi
710 }

```

`\@disable@onlypremakeg` Disable all commands listed in `\@gls@onlypremakeg`

```

711 \newcommand*\@disable@onlypremakeg{%
712 \@for\@thiscs:=\@gls@onlypremakeg\do{%
713   \expandafter\@disable@premakecs\@thiscs%
714 }}

```

`\@disable@premakecs` Disables the given command.

```

715 \newcommand*\@disable@premakecs[1]{%
716   \def#1{\PackageError{glossaries}{\string#1\space may only be
717   used before \string\makeglossaries}{You can't use
718   \string#1\space after \string\makeglossaries}}%
719 }

```

1.3 Default values

This section sets up default values that are used by this package. Some of the names may already be defined (e.g. by) so `\providecommand` is used.

Main glossary title:

`\glossaryname`

```
720 \providecommand*{\glossaryname}{Glossary}
```

The title for the acronym glossary type (which is defined if acronym package option is used) is given by `\acronymname`. If the acronym package option is not used, `\acronymname` won't be used.

`\acronymname`

```
721 \providecommand*{\acronymname}{Acronyms}
```

`\glssettoctitle` Sets the TOC title for the given glossary.

```
722 \newcommand*{\glssettoctitle}[1]{%
```

```
723 \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}
```

The following commands provide text for the headers used by some of the tabular-like glossary styles. Whether or not they get used in the glossary depends on the glossary style.

`\entryname`

```
724 \providecommand*{\entryname}{Notation}
```

`\descriptionname`

```
725 \providecommand*{\descriptionname}{Description}
```

`\symbolname`

```
726 \providecommand*{\symbolname}{Symbol}
```

`\pagelistname`

```
727 \providecommand*{\pagelistname}{Page List}
```

Labels for `makeindex`'s symbol and number groups:

`glsymbolsgroupname`

```
728 \providecommand*{\glsymbolsgroupname}{Symbols}
```

`glsnumbersgroupname`

```
729 \providecommand*{\glsnumbersgroupname}{Numbers}
```

`\glspluralsuffix` The default plural is formed by appending `\glspluralsuffix` to the singular form.

```
730 \newcommand*{\glspluralsuffix}{s}
```

```
\seename
731 \providecommand*\seename}{see}
```

```
\andname
732 \providecommand*\andname}{\&}
```

Add multi-lingual support. Thanks to everyone who contributed to the translations from both comp.text.tex and via email.

`\addglossarytocaptions` If using, `\glossaryname` should be defined in terms of `\translate`, but if `babel` is also loaded, it will redefine `\glossaryname` whenever the language is set, so override it. (Don't use `\addto` as doesn't define it.)

```
733 \newcommand*\addglossarytocaptions}[1]{%
734   \ifcsundef{captions#1}{}%
735   {%
736     \expandafter\let\expandafter\@gls@tmp\csname captions#1\endcsname
737     \expandafter\toks@\expandafter{\@gls@tmp
738       \renewcommand*\glossaryname}{\translate{Glossary}}%
739     }%
740     \expandafter\edef\csname captions#1\endcsname{\the\toks@}%
741   }%
742 }
```

```
743 \ifglstranslate
```

If is not install, used standard captions, otherwise load dictionary.

```
744 \@ifpackageloaded{translator}{%
745   \usedictionary{glossaries-dictionary}%
746   \addglossarytocaptions{portuges}%
747   \addglossarytocaptions{portuguese}%
748   \addglossarytocaptions{brazil}%
749   \addglossarytocaptions{brazilian}%
750   \addglossarytocaptions{danish}%
751   \addglossarytocaptions{dutch}%
752   \addglossarytocaptions{afrikaans}%
753   \addglossarytocaptions{english}%
754   \addglossarytocaptions{UKenglish}%
755   \addglossarytocaptions{USenglish}%
756   \addglossarytocaptions{american}%
757   \addglossarytocaptions{australian}%
758   \addglossarytocaptions{british}%
759   \addglossarytocaptions{canadian}%
760   \addglossarytocaptions{newzealand}%
761   \addglossarytocaptions{french}%
762   \addglossarytocaptions{frenchb}%
763   \addglossarytocaptions{français}%
764   \addglossarytocaptions{acadian}%
765   \addglossarytocaptions{canadien}%
766   \addglossarytocaptions{german}%
```

```

767 \addglossarytocaptions{germanb}%
768 \addglossarytocaptions{austrian}%
769 \addglossarytocaptions{naustrian}%
770 \addglossarytocaptions{ngerman}%
771 \addglossarytocaptions{irish}%
772 \addglossarytocaptions{italian}%
773 \addglossarytocaptions{magyar}%
774 \addglossarytocaptions{hungarian}%
775 \addglossarytocaptions{polish}%
776 \addglossarytocaptions{spanish}%
777 \renewcommand*{\glssettoctitle}[1]{%
778   \ifcsdef{gls@tr@set@#1@toctitle}%
779     {%
780       \csuse{gls@tr@set@#1@toctitle}%
781     }%
782   {%
783     \def\glossarytoctitle{\csname @glotype@#1@title\endcsname}%
784   }%
785 }%
786 \renewcommand*{\glossaryname}{\translate{Glossary}}%
787 \renewcommand*{\acronymname}{\translate{Acronyms}}%
788 \renewcommand*{\entryname}{\translate{Notation (glossaries)}}%
789 \renewcommand*{\descriptionname}{%
790   \translate{Description (glossaries)}}%
791 \renewcommand*{\symbolname}{\translate{Symbol (glossaries)}}%
792 \renewcommand*{\pagelistname}{%
793   \translate{Page List (glossaries)}}%
794 \renewcommand*{\glssymbolsgroupname}{%
795   \translate{Symbols (glossaries)}}%
796 \renewcommand*{\glsnumbersgroupname}{%
797   \translate{Numbers (glossaries)}}%
798 }{%

799 \@ifpackageloaded{polyglossia}%
800 { \RequirePackage{glossaries-polyglossia}}%
801 {%
802   \@ifpackageloaded{babel}{%
803     \RequirePackage{glossaries-babel}}{%
804   }}
805 \else

806 \@gls@notranslatorhook
807 \fi

```

`\nopostdesc` Provide a means to suppress description terminator for a given entry. (Useful for entries with no description.) Has no effect outside the glossaries.

```
808 \DeclareRobustCommand*{\nopostdesc}{}
```

`\@nopostdesc` Suppress next description terminator.


```

809 \newcommand*\@nopostdesc}{%
810   \let\org@glspostdescription\glspostdescription
811   \def\glspostdescription{%
812     \let\glspostdescription\org@glspostdescription}%
813 }

```

`\@no@post@desc` Used for comparison purposes.

```
814 \newcommand*\@no@post@desc}{\nopostdesc}
```

`\glspar` Provide means of having a paragraph break in glossary entries

```
815 \newcommand{\glspar}{\par}
```

`\setStyleFile` Sets the style file. The relevant extension is appended.

```

816 \newcommand{\setStyleFile}[1]{%
817   \renewcommand*\gl@istfilebase{#1}%
   Just in case \istfilename has been modified.
818   \ifglxindy
819     \def\istfilename{\gl@istfilebase.xdy}
820   \else
821     \def\istfilename{\gl@istfilebase.ist}
822   \fi
823 }

```

This command only has an effect prior to using `\makeglossaries`.

```
824 \@onlypremakeg\setStyleFile
```

The name of the `makeindex` or `xindy` style file is given by `\istfilename`. This file is created by `\writeist` (which is used by `\makeglossaries`) so redefining this command will only have an effect if it is done *before* `\makeglossaries`. As from v1.17, use `\setStyleFile` instead of directly redefining `\istfilename`.

`\istfilename`

```

825 \ifglxindy
826   \def\istfilename{\gl@istfilebase.xdy}
827 \else
828   \def\istfilename{\gl@istfilebase.ist}
829 \fi

```

`\gl@istfilebase`

```
830 \newcommand*\gl@istfilebase}{\jobname}
```

The `makeglossaries` Perl script picks up this name from the auxiliary file. If the name ends with `.xdy` it calls `xindy` otherwise it calls `makeindex`. Since its not required by `LATEX`, `\@istfilename` ignores its argument.

`\@istfilename`

```
831 \newcommand*\@istfilename}[1]{}

```

This command is the value of the `page_compositor` `makeindex` key. Again, any redefinition of this command must take place *before* `\writeist` otherwise it will have no effect. As from 1.17, use `\glsSetCompositor` instead of directly redefining `\glscompositor`.

`\glscompositor`

```
832 \newcommand*\glscompositor}{.}
```

`\glsSetCompositor` Sets the compositor.

```
833 \newcommand*\glsSetCompositor[1]{%
834   \renewcommand*\glscompositor{#1}}
```

Only use before `\makeglossaries`

```
835 \@onlypremakeg\glsSetCompositor
```

(The page compositor is usually defined as a dash when using `makeindex`, but most of the standard counters used by \TeX use a full stop as the compositor, which is why I have used it as the default.) If `xindy` is used `\glscompositor` only affects the `arabic-page-numbers` location class.

`@glsAlphacompositor` This is only used by `xindy`. It specifies the compositor to use when location numbers are in the form `<letter><compositor><number>`. For example, if `@glsAlphacompositor` is set to `."` then it allows locations such as `A.1` whereas if `@glsAlphacompositor` is set to `-"` then it allows locations such as `A-1`.

```
836 \newcommand*\@glsAlphacompositor{\glscompositor}
```

`sSetAlphaCompositor` Sets the alpha compositor.

```
837 \ifglxindy
838   \newcommand*\glsSetAlphaCompositor[1]{%
839     \renewcommand*\@glsAlphacompositor{#1}}
840 \else
841   \newcommand*\glsSetAlphaCompositor[1]{%
842     \glsnoxindywarning\glsSetAlphaCompositor}
843 \fi
```

Can only be used before `\makeglossaries`

```
844 \@onlypremakeg\glsSetAlphaCompositor
```

`\gls@suffixF` Suffix to use for a two page list. This overrides the separator and the closing page number if set to something other than an empty macro.

```
845 \newcommand*\gls@suffixF}{}
```

`\glsSetSuffixF` Sets the suffix to use for a two page list.

```
846 \newcommand*\glsSetSuffixF[1]{%
847   \renewcommand*\gls@suffixF{#1}}
```

Only has an effect when used before `\makeglossaries`

```
848 \@onlypremakeg\glsSetSuffixF
```

`\gls@suffixFF` Suffix to use for a three page list. This overrides the separator and the closing page number if set to something other than an empty macro.

```
849 \newcommand*\gls@suffixFF{}
```

`\glsSetSuffixFF` Sets the suffix to use for a three page list.

```
850 \newcommand*\glsSetSuffixFF[1]{%
851   \renewcommand*\gls@suffixFF{#1}%
852 }
```

`\glsnumberformat` The command `\glsnumberformat` indicates the default format for the page numbers in the glossary. (Note that this is not the same as `\glossaryentrynumbers`, but applies to individual numbers or groups of numbers within an entry's associated number list.) If hyperlinks are defined, it will use `\glsnumber`, otherwise it will simply display its argument “as is”.

```
853 \ifcsundef{hyperlink}%
854 {%
855   \newcommand*\glsnumberformat[1]{#1}%
856 }%
857 {%
858   \newcommand*\glsnumberformat[1]{\glsnumber{#1}}%
859 }
```

Individual numbers in an entry's associated number list are delimited using `\delimN` (which corresponds to the `delim_n makeindex` keyword). The default value is a comma followed by a space.

`\delimN`

```
860 \newcommand{\delimN}{, }
```

A range of numbers within an entry's associated number list is delimited using `\delimR` (which corresponds to the `delim_r makeindex` keyword). The default is an en-dash.

`\delimR`

```
861 \newcommand{\delimR}{--}
```

The glossary preamble is given by `\glossarypreamble`. This will appear after the glossary sectioning command, and before the `\glossary` environment. It is designed to allow the user to add information pertaining to the glossary (e.g. “page numbers in italic indicate the primary definition”) therefore `\glossarypreamble` shouldn't be affected by the glossary style. (So if you define your own glossary style, don't have it change `\glossarypreamble`.) The preamble is empty by default. If you have multiple glossaries, and you want a different preamble for each glossary, you will need to use `\printglossary` for each glossary type, instead of `\printglossaries`, and redefine `\glossarypreamble` before each `\printglossary`.

`\glossarypreamble`

```
862 \newcommand*\glossarypreamble}{%
863   \csuse{@glossarypreamble@\currentglossary}%
864 }
```

`\setglossarypreamble` `\setglossarypreamble[<type>]{<text>}`

Code provided by Michael Pock.

```
865 \newcommand\setglossarypreamble}[2] [\glsdefaulttype]{%
866   \ifglossaryexists{#1}{%
867     \csgdef{@glossarypreamble@#1}{#2}%
868   }{%
869     \GlossariesWarning{%
870       Glossary ‘#1’ is not defined%
871     }%
872   }%
873 }
```

The glossary postamble is given by `\glossarypostamble`. This is provided to allow the user to add something after the end of the `theglossary` environment (again, this shouldn't be affected by the glossary style). It is, of course, possible to simply add the text after `\printglossary`, but if you only want the postamble to appear after the first glossary, but not after subsequent glossaries, you can do something like:

```
\renewcommand{\glossarypostamble}{For a complete list of terms
see \cite{blah}\gdef\glossarypreamble{}}
```

`\glossarypostamble`

```
874 \newcommand*\glossarypostamble}{}
```

`\glossarysection`

The sectioning command that starts a glossary is given by `\glossarysection`. (This does not form part of the glossary style, and so should not be changed by a glossary style.) If `\phantomsection` is defined, it uses `\p@glossarysection`, otherwise it uses `\@glossarysection`.

```
875 \newcommand*\glossarysection}[2] [\@gls@title]{%
876   \def\@gls@title{#2}%
877   \ifcsundef{phantomsection}%
878     {%
879       \@glossarysection{#1}{#2}%
880     }%
881     {%
882       \p@glossarysection{#1}{#2}%
883     }%
884   \gls@glossarymark{\glossarytoctitle}%
885 }
```

`\glsglossarymark` Sets the header mark for the glossary. Takes the glossary short (TOC) title as the argument.

```
886 \ifcsundef{glossarymark}%
887 {%
888   \newcommand{\glsglossarymark}[1]{\glossarymark{#1}}
889 }%
890 {%
891   \@ifclassloaded{memoir}
892   {%
893     \newcommand{\glsglossarymark}[1]{%
894       \ifglsucmark
895         \markboth{\memUHead{#1}}{\memUHead{#1}}%
896       \else
897         \markboth{#1}{#1}%
898       \fi
899     }
900   }%
901   {%
902     \newcommand{\glsglossarymark}[1]{%
903       \ifglsucmark
904         \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
905       \else
906         \@mkboth{#1}{#1}%
907       \fi
908     }
909   }
910 }
```

`\glossarymark` Provided for backward compatibility:

```
911 \providecommand{\glossarymark}[1]{%
912   \ifglsucmark
913     \@mkboth{\mfirstucMakeUppercase{#1}}{\mfirstucMakeUppercase{#1}}%
914   \else
915     \@mkboth{#1}{#1}%
916   \fi
917 }
```

The required sectional unit is given by `\@@glossarysec` which was defined by the section package option. The starred form of the command is chosen. If you don't want any sectional command, you will need to redefine `\glossarysection`. The sectional unit can be changed, if different sectional units are required.

`\setglossarysection`

```
918 \newcommand*\setglossarysection[1]{%
919 \setkeys{glossaries.sty}{section=#1}}
```

The command `\@glossarysection` indicates how to start the glossary section if `\phantomsection` is not defined.

`\@glossarysection`

```
920 \newcommand*\@glossarysection}[2]{%
921   \ifdefempty\@glossarysecstar
922   {%
923     \csname\@glossarysec\endcsname[#1]{#2}%
924   }%
925   {%
926     \csname\@glossarysec\endcsname*{#2}%
927     \@gls@toc{#1}{\@glossarysec}%
928   }%
```

Do automatic labelling if required

```
929   \@glossaryseclabel
930 }
```

As `\@glossarysection`, but put in `\phantomsection`, and swap where `\@gls@toc` goes. If using chapters do a `\clearpage`. This ensures that the hyper link from the table of contents leads to the line above the heading, rather than the line below it.

`\@p@glossarysection`

```
931 \newcommand*\@p@glossarysection}[2]{%
932   \gls@clearpage
933   \phantomsection
934   \ifdefempty\@glossarysecstar
935   {%
936     \csname\@glossarysec\endcsname{#2}%
937   }%
938   {%
939     \@gls@toc{#1}{\@glossarysec}%
940     \csname\@glossarysec\endcsname*{#2}%
941   }%
```

Do automatic labelling if required

```
942   \@glossaryseclabel
943 }
```

`\gls@docclearpage` The `\gls@docclearpage` command is used to issue a `\clearpage` (or `\cleardoublepage`) depending on whether the glossary sectional unit is a chapter. If the sectional unit is something else, do nothing.

```
944 \newcommand*\gls@docclearpage{%
945   \ifthenelse{\equal{\@glossarysec}{chapter}}%
946   {%
947     \ifcsundef{cleardoublepage}%
948     {%
949       \clearpage
950     }%
951   }%
952   \ifcsdef{if@openright}%
```

```

953     {%
954     \if@openright
955         \cleardoublepage
956     \else
957         \clearpage
958     \fi
959     }%
960     {%
961     \cleardoublepage
962     }%
963     }%
964 }%
965 {}%
966 }

```

`\glsclearpage` This just calls `\gls@doclearpage`, but it makes it easier to have a user command so that the user can override it.

```

967 \newcommand*\glsclearpage{\gls@doclearpage}

```

The glossary is added to the table of contents if `glstoc` flag set. If it is set, `\@gls@toc` will add a line to the `.toc` file, otherwise it will do nothing. (The first argument to `\@gls@toc` is the title for the table of contents, the second argument is the sectioning type.)

`\@gls@toc`

```

968 \newcommand*\@gls@toc[2]{%
969     \ifglstoc
970         \ifglsnumberline
971             \addcontentsline{toc}{#2}{\numberline{#1}}%
972         \else
973             \addcontentsline{toc}{#2}{#1}%
974         \fi
975     \fi
976 }

```

1.4 Xindy

This section defines commands that only have an effect if `xindy` is used to sort the glossaries.

`\glsnoxindywarning` Issues a warning if `xindy` hasn't been specified. These warnings can be suppressed by redefining `\glsnoxindywarning` to ignore its argument

```

977 \newcommand*\glsnoxindywarning[1]{%
978     \GlossariesWarning{Not in xindy mode --- ignoring \string#1}%
979 }

```

`\@xdyattributes` Define list of attributes (`\string` is used in case the double quote character has been made active)

```

980 \ifglxsindy
981   \edef\@xdyattributes{\string"default\string"}%
982 \fi

```

`\@xdyattributelist` Comma-separated list of attributes.

```

983 \ifglxsindy
984   \edef\@xdyattributelist{}%
985 \fi

```

`\@xdylocref` Define list of markup location references.

```

986 \ifglxsindy
987   \def\@xdylocref{}
988 \fi

```

`\@gls@ifinlist`

```

989 \newcommand*\@gls@ifinlist[4]{%
990   \def\@do@ifinlist##1,#1,##2\end@do@ifinlist{%
991     \def\@gls@listsuffix{##2}%
992     \ifx\@gls@listsuffix\@empty
993       #4%
994     \else
995       #3%
996     \fi
997   }%
998   \@do@ifinlist,#2,#1,\end@do@ifinlist
999 }

```

`\GlsAddXdyCounters` Need to know all the counters that will be used in location numbers for Xindy. Argument may be a single counter name or a comma-separated list of counter names.

```

1000 \ifglxsindy
1001   \newcommand*\@xdycounters{\glscounter}
1002   \newcommand*\GlsAddXdyCounters[1]{%
1003     \@for\@gls@ctr:=#1\do{%
1004       \edef\@do@addcounter{%
1005         \noexpand\@gls@ifinlist{\@gls@ctr}{\@xdycounters}{}%
1006         {%
1007           \noexpand\edef\noexpand\@xdycounters{\@xdycounters,%
1008             \noexpand\@gls@ctr}%
1009         }%
1010       }%
1011       \@do@addcounter
1012     }
1013   }

```

Only has an effect before `\writeist`:

```

1014   \@onlypremakeg\GlsAddXdyCounters

```



```

1015 \else
1016   \newcommand*\GlsAddXdyCounters[1]{%
1017     \glsnoxywarning\GlsAddXdyAttribute
1018   }
1019 \fi

```

`@glsaddxdycounters` Counters must all be identified before adding attributes.

```

1020 \newcommand*\@disabled@glsaddxdycounters{%
1021   \PackageError{glossaries}{\string\GlsAddXdyCounters\space
1022   can't be used after \string\GlsAddXdyAttribute}{Move all
1023   occurrences of \string\GlsAddXdyCounters\space before the first
1024   instance of \string\GlsAddXdyAttribute}%
1025 }

```

`\GlsAddXdyAttribute` Adds an attribute.

```

1026 \ifglxindy

```

First define internal command that adds an attribute for a given counter (2nd argument is the counter):

```

1027 \newcommand*\@glsaddxdyattribute[2]{%

```

Add to xindy attribute list

```

1028   \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string" ^^J
1029     \string"#2#1\string"}%

```

Add to xindy markup location.

```

1030   \expandafter\toks@\expandafter{\@xdylocref}%
1031   \edef\@xdylocref{\the\toks@ ^^J%
1032     (markup-locref
1033     :open \string"~\glstildechar n%
1034     \expandafter\string\csname glsX#2X#1\endcsname
1035     \string" ^^J
1036     :close \string"\string" ^^J
1037     :attr \string"#2#1\string")}%

```

Define associated attribute command `\glsX<counter>X<attribute>{\Hprefix}\{<n>}`

```

1038   \expandafter\gdef\csname glsX#2X#1\endcsname##1##2{%
1039     \setentrycounter{##1}{#2}\csname #1\endcsname{##2}%
1040   }%
1041 }

```

High-level command:

```

1042 \newcommand*\GlsAddXdyAttribute[1]{%

```

Add to comma-separated attribute list

```

1043   \ifx\@xdyattributelist\@empty
1044     \edef\@xdyattributelist{#1}%
1045   \else
1046     \edef\@xdyattributelist{\@xdyattributelist,#1}%
1047   \fi

```

Iterate through all specified counters and add counter-dependent attributes:

```
1048 \for\@this@counter:=\@xdycounters\do{%
1049 \protected@edef\gls@do@addxdyattribute{%
1050 \noexpand\@glsaddxdyattribute{#1}{\@this@counter}%
1051 }
1052 \gls@do@addxdyattribute
1053 }%
```

All occurrences of `\GlsAddXdyCounters` must be used before this command

```
1054 \let\GlsAddXdyCounters\@disabled@glsaddxdycounters
1055 }
```

Only has an effect before `\writeist`:

```
1056 \@onlypremakeg\GlsAddXdyAttribute
1057 \else
1058 \newcommand*\GlsAddXdyAttribute[1]{%
1059 \glsnoxindywarning\GlsAddXdyAttribute}
1060 \fi
```

`redefinedattributes` Add known attributes for all defined counters

```
1061 \ifglsxindy
1062 \newcommand*\@gls@addpredefinedattributes}{%
1063 \GlsAddXdyAttribute{glsnumberformat}
1064 \GlsAddXdyAttribute{textrm}
1065 \GlsAddXdyAttribute{textsf}
1066 \GlsAddXdyAttribute{texttt}
1067 \GlsAddXdyAttribute{textbf}
1068 \GlsAddXdyAttribute{textmd}
1069 \GlsAddXdyAttribute{textit}
1070 \GlsAddXdyAttribute{textup}
1071 \GlsAddXdyAttribute{textsl}
1072 \GlsAddXdyAttribute{textsc}
1073 \GlsAddXdyAttribute{emph}
1074 \GlsAddXdyAttribute{glshypernumber}
1075 \GlsAddXdyAttribute{hyperrrm}
1076 \GlsAddXdyAttribute{hypersf}
1077 \GlsAddXdyAttribute{hypertt}
1078 \GlsAddXdyAttribute{hyperbf}
1079 \GlsAddXdyAttribute{hypermd}
1080 \GlsAddXdyAttribute{hyperit}
1081 \GlsAddXdyAttribute{hyperup}
1082 \GlsAddXdyAttribute{hypersl}
1083 \GlsAddXdyAttribute{hypersc}
1084 \GlsAddXdyAttribute{hyperemph}
1085 }
1086 \else
1087 \let\@gls@addpredefinedattributes\relax
1088 \fi
```

`\@xdyuseralphabets` List of additional alphabets

```
1089 \def\@xdyuseralphabets{}
```

`\GlsAddXdyAlphabet` `\GlsAddXdyAlphabet{<name>}{<definition>}` adds a new alphabet called `<name>`.
The definition must use xindy syntax.

```
1090 \ifglxsindy
1091   \newcommand*{\GlsAddXdyAlphabet}[2]{%
1092     \edef\@xdyuseralphabets{%
1093       \@xdyuseralphabets ^^J
1094       (define-alphabet "#1" (#2))}}
1095 \else
1096   \newcommand*{\GlsAddXdyAlphabet}[2]{%
1097     \glsnnoxindywarning\GlsAddXdyAlphabet}
1098 \fi
```

This code is only required for xindy:

```
1099 \ifglxsindy
```

`\@xdy@locationlist` List of predefined location names.

```
1100 \newcommand*{\@gls@xdy@locationlist}{%
1101   roman-page-numbers,%
1102   Roman-page-numbers,%
1103   arabic-page-numbers,%
1104   alpha-page-numbers,%
1105   Alpha-page-numbers,%
1106   Appendix-page-numbers,%
1107   arabic-section-numbers%
1108 }
```

Each location class `<name>` has the format stored in `\@gls@xdy@Lclass@<name>`.
Set up predefined formats.

`\@roman-page-numbers` Lower case Roman numerals (i, ii, ...). In the event that `\roman` has been redefined to produce a fancy form of roman numerals, attempt to work out how it will be written to the output file.

```
1109 \protected@edef\@gls@roman{\@roman{0}\string"
1110   \string"roman-numbers-lowercase\string" :sep \string"}%
1111 \@onelevel@sanitize\@gls@roman
1112 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
1113   :sep \string"}%
1114 \@onelevel@sanitize\@tmp
1115 \ifx\@tmp\@gls@roman
1116   \expandafter
1117     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{%
1118       \string"roman-numbers-lowercase\string"%
1119     }%
1120 \else
1121   \expandafter
1122     \edef\csname @gls@xdy@Lclass@roman-page-numbers\endcsname{
```

```

1123         :sep \string"\@gls@roman\string"%
1124     }%
1125 \fi

@Roman-page-numbers Upper case Roman numerals (I, II, ...).
1126 \expandafter\def\csname @gls@xdy@Lclass@Roman-page-numbers\endcsname{%
1127     \string"roman-numbers-uppercase\string"%
1128 }%

arabic-page-numbers Arabic numbers (1, 2, ...).
1129 \expandafter\def\csname @gls@xdy@Lclass@arabic-page-numbers\endcsname{%
1130     \string"arabic-numbers\string"%
1131 }%

@alpha-page-numbers Lower case alphabetical (a, b, ...).
1132 \expandafter\def\csname @gls@xdy@Lclass@alpha-page-numbers\endcsname{%
1133     \string"alpha\string"%
1134 }%

@Alpha-page-numbers Upper case alphabetical (A, B, ...).
1135 \expandafter\def\csname @gls@xdy@Lclass@Alpha-page-numbers\endcsname{%
1136     \string"ALPHA\string"%
1137 }%

appendix-page-numbers Appendix style locations (e.g. A-1, A-2, ..., B-1, B-2, ...). The separator is given
by \@glsAlphacompositor.
1138 \expandafter\def\csname @gls@xdy@Lclass@Appendix-page-numbers\endcsname{%
1139     \string"ALPHA\string"
1140     :sep \string"\@glsAlphacompositor\string"
1141     \string"arabic-numbers\string"%
1142 }

arabic-section-numbers Section number style locations (e.g. 1.1, 1.2, ...). The compositor is given by
\glscompositor.
1143 \expandafter\def\csname @gls@xdy@Lclass@arabic-section-numbers\endcsname{%
1144     \string"arabic-numbers\string"
1145     :sep \string"\glscompositor\string"
1146     \string"arabic-numbers\string"%
1147 }%

xdyuserlocationdefs List of additional location definitions (separated by ^^J)
1148 \def\@xdyuserlocationdefs{}

xdyuserlocationnames List of additional user location names
1149 \def\@xdyuserlocationnames{}

End of xindy-only block:
1150 \fi

```

`\GlsAddXdyLocation` `\GlsAddXdyLocation[<prefix-loc>]{<name>}{<definition>}` Define a new location called *<name>*. The definition must use xindy syntax. (Note that this doesn't check to see if the location is already defined. That is left to xindy to complain about.)

```

1151 \ifglxindy
1152   \newcommand*\GlsAddXdyLocation[3][{}]{%
1153     \def\@gls@tmp{#1}%
1154     \ifx\@gls@tmp\@empty
1155       \edef\@xdyuserlocationdefs{%
1156         \@xdyuserlocationdefs ^^J%
1157         (define-location-class \string"#2\string"^^J\space\space
1158         \space(:sep \string"{} \glsopenbrace\string" #3
1159           :sep \string"\glsclosebrace\string"))
1160       }%
1161     \else
1162       \edef\@xdyuserlocationdefs{%
1163         \@xdyuserlocationdefs ^^J%
1164         (define-location-class \string"#2\string"^^J\space\space
1165         \space(:sep "\glsopenbrace"
1166           #1
1167           :sep "\glsclosebrace\glsopenbrace" #3
1168           :sep "\glsclosebrace"))
1169       }%
1170     \fi
1171     \edef\@xdyuserlocationnames{%
1172       \@xdyuserlocationnames^^J\space\space\space
1173       \string"#1\string"}%
1174   }

```

Only has an effect before `\writeist`:

```

1175 \@onlypremakeg\GlsAddXdyLocation
1176 \else
1177   \newcommand*\GlsAddXdyLocation[2]{%
1178     \glsnoxindywarning\GlsAddXdyLocation}
1179 \fi

```

`\locationclassorder` Define location class order

```

1180 \ifglxindy
1181   \edef\@xdylocationclassorder{^^J\space\space\space
1182     \string"roman-page-numbers\string"^^J\space\space\space
1183     \string"arabic-page-numbers\string"^^J\space\space\space
1184     \string"arabic-section-numbers\string"^^J\space\space\space
1185     \string"alpha-page-numbers\string"^^J\space\space\space
1186     \string"Roman-page-numbers\string"^^J\space\space\space
1187     \string"Alpha-page-numbers\string"^^J\space\space\space
1188     \string"Appendix-pages-numbers\string"
1189     \@xdyuserlocationnames^^J\space\space\space
1190     \string"see\string"
1191   }

```

```
1192 \fi
```

Change the location order.

yLocationClassOrder

```
1193 \ifglxindy
1194   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
1195     \def\@xdylocationclassorder{#1}}
1196 \else
1197   \newcommand*\GlsSetXdyLocationClassOrder[1]{%
1198     \glsnoxywarning\GlsSetXdyLocationClassOrder}
1199 \fi
```

\@xdysortrules Define sort rules

```
1200 \ifglxindy
1201   \def\@xdysortrules{}
1202 \fi
```

\GlsAddSortRule Add a sort rule

```
1203 \ifglxindy
1204   \newcommand*\GlsAddSortRule[2]{%
1205     \expandafter\toks@\expandafter{\@xdysortrules}%
1206     \protected@edef\@xdysortrules{\the\toks@ ^^J
1207       (sort-rule \string"#1\string" \string"#2\string")}%
1208   }
1209 \else
1210   \newcommand*\GlsAddSortRule[2]{%
1211     \glsnoxywarning\GlsAddSortRule}
1212 \fi
```

\@xdyrequiredstyles Define list of required styles (this should be a comma-separated list of xindy styles)

```
1213 \ifglxindy
1214   \def\@xdyrequiredstyles{tex}
1215 \fi
```

\GlsAddXdyStyle Add a xindy style to the list of required styles

```
1216 \ifglxindy
1217   \newcommand*\GlsAddXdyStyle[1]{%
1218     \edef\@xdyrequiredstyles{\@xdyrequiredstyles,#1}}%
1219 \else
1220   \newcommand*\GlsAddXdyStyle[1]{%
1221     \glsnoxywarning\GlsAddXdyStyle}
1222 \fi
```

\GlsSetXdyStyles Reset the list of required styles

```
1223 \ifglxindy
1224   \newcommand*\GlsSetXdyStyles[1]{%
```

```

1225   \edef\@xdyrequiredstyles{#1}}
1226 \else
1227   \newcommand*\GlsSetXdyStyles[1]{%
1228     \glsnoxindywarning\GlsSetXdyStyles}
1229 \fi

```

`\findrootlanguage` This used to determine the root language, using a bit of trickery since babel doesn't supply the information, but now that babel is once again actively maintained, we can't do this any more, so `\findrootlanguage` is no longer available. Now provide a command that does nothing (in case it's been patched), but this may be removed completely in the future.

```

1230 \newcommand*\findrootlanguage{}

```

`\@xdylanguage` The xindy language setting is required by `makeglossaries`, so provide a command for `makeglossaries` to pick up the information from the auxiliary file. This command is not needed by the glossaries package, so define it to ignore its arguments.

```

1231 \def\@xdylanguage#1#2{}

```

`\GlsSetXdyLanguage` Define a command that allows the user to set the language for a given glossary type. The first argument indicates the glossary type. If omitted the main glossary is assumed.

```

1232 \ifglxindy
1233   \newcommand*\GlsSetXdyLanguage[2][\glsdefaulttype]{%
1234     \ifglossaryexists{#1}{%
1235       \expandafter\def\csname @xdy@#1@language\endcsname{#2}%
1236     }{%
1237       \PackageError{glossaries}{Can't set language type for
1238         glossary type '#1' --- no such glossary}{%
1239         You have specified a glossary type that doesn't exist}}
1240 \else
1241   \newcommand*\GlsSetXdyLanguage[2][]{%
1242     \glsnoxindywarning\GlsSetXdyLanguage}
1243 \fi

```

`\@gls@codepage` The xindy codepage setting is required by `makeglossaries`, so provide a command for `makeglossaries` to pick up the information from the auxiliary file. This command is not needed by the glossaries package, so define it to ignore its arguments.

```

1244 \def\@gls@codepage#1#2{}

```

`\GlsSetXdyCodePage` Define command to set the code page.

```

1245 \ifglxindy
1246   \newcommand*\GlsSetXdyCodePage[1]{%
1247     \renewcommand*\@gls@codepage{#1}%
1248   }

```

Suggested by egreg:

```
1249 \AtBeginDocument{%
1250   \ifx\gls@codepage\@empty
1251     \@ifpackageloaded{fontspec}{\def\gls@codepage{utf8}}{}%
1252   \fi
1253 }
1254 \else
1255   \newcommand*\{GlsSetXdyCodePage}[1]{%
1256     \glsnoxindywarning\GlsSetXdyCodePage}
1257 \fi
```

`\@xdylettergroups` Store letter group definitions.

```
1258 \ifglxindy
1259   \ifglsexindy@glnumbers
1260     \def\@xdylettergroups{(define-letter-group
1261       \string\glnumbers\string^^J\space\space\space
1262       :prefixes (\string"0\string" \string"1\string"
1263       \string"2\string" \string"3\string" \string"4\string"
1264       \string"5\string" \string"6\string" \string"7\string"
1265       \string"8\string" \string"9\string")^^J\space\space\space
1266       :before \string"@glsfirstletter\string")}
1267   \else
1268     \def\@xdylettergroups{}
1269   \fi
1270 \fi
```

`\GlsAddLetterGroup` Add a new letter group. The first argument is the name of the letter group. The second argument is the xindy code specifying prefixes and ordering.

```
1271 \newcommand*\GlsAddLetterGroup[2]{%
1272   \expandafter\toks@\expandafter{\@xdylettergroups}%
1273   \protected@edef\@xdylettergroups{\the\toks@^^J%
1274   (define-letter-group \string"#1\string"^^J\space\space\space#2)}%
1275 }
```

1.5 Loops and conditionals

`\forallglossaries` To iterate through all glossaries (or comma-separated list of glossary names given in optional argument) use:

```
\forallglossaries[<glossary list>]{<cmd>}{<code>}
```

where *<cmd>* is a control sequence which will be set to the name of the glossary in the current iteration.

```
1276 \newcommand*\forallglossaries[3][\@glo@types]{%
1277   \@for#2:=#1\do{\ifx#2\@empty\else#3\fi}%
1278 }
```

`\forallacronyms`


```

1279 \newcommand*\forallacronyms}[2]{%
1280   \@for#1:=\@glsacronymlists\do{\ifx#1\@empty\else#2\fi}%
1281 }

```

`\forglentries` To iterate through all entries in a given glossary use:

```
\forglentries[<type>]{<cmd>}{<code>}
```

where *<type>* is the glossary label and *<cmd>* is a control sequence which will be set to the entry label in the current iteration.

```

1282 \newcommand*\forglentries}[3][\glsdefaulttype]{%
1283   \edef\@glo@list{\csname glolist@#1\endcsname}%
1284   \@for#2:=\@glo@list\do
1285   {%
1286     \ifdefempty{#2}{#3}%
1287   }%
1288 }

```

`\forallglsentries` To iterate through all glossary entries over all glossaries listed in the optional argument (the default is all glossaries) use:

```
\forallglsentries[<glossary list>]{<cmd>}{<code>}
```

Within `\forallglsentries`, the current glossary type is given by `\@thisglo@`.

```

1289 \newcommand*\forallglsentries}[3][\@glo@types]{%
1290   \expandafter\forallglossaries\expandafter[#1]{\@thisglo@}%
1291   {%
1292     \forglentries[\@thisglo@]{#2}{#3}%
1293   }%
1294 }

```

`\ifglossaryexists` To check to see if a glossary exists use:

```
\ifglossaryexists{<type>}{<true-text>}{<false-text>}
```

where *<type>* is the glossary's label.

```

1295 \newcommand{\ifglossaryexists}[3]{%
1296   \ifcsundef{@glo@type@#1@out}{#3}{#2}%
1297 }

```

Since the label is used to form the name of control sequences, by default UTF8 etc characters can't be used in the label. A possible workaround is to use `\scantokens`, but commands such as `\glsentrytext` will no longer be usable in sectioning, caption etc commands. If the user really wants to be able to construct a label with UTF8 characters, allow them the means to do so (but on their own head be it, if they then use entries in `\section` etc). This can be done via:

```
\renewcommand*\glsdetoklabel}[1]{\scantokens{#1\noexpand}}
```

(Note, don't use `\detokenize` or it will cause commands like `\glsaddall` to fail.) Since redefining `\glsdetoklabel` can cause things to go badly wrong, I'm not going to mention it in the main user guide. Only advanced users who know what they're doing ought to attempt it.

`\glsdetoklabel`

```
1298 \newcommand*\glsdetoklabel[1]{#1}
```

`\ifglsentryexists` To check to see if a glossary entry has been defined use:

```
\ifglsentryexists{<label>}{<true text>}{<false text>}
```

where `<label>` is the entry's label.

```
1299 \newcommand{\ifglsentryexists}[3]{%
1300   \ifcsundef{glo@\glsdetoklabel{#1}@name}{#3}{#2}%
1301 }
```

`\ifglsused` To determine if given glossary entry has been used in the document text yet use:

```
\ifglsused{<label>}{<true text>}{<false text>}
```

where `<label>` is the entry's label. If true it will do `<true text>` otherwise it will do `<false text>`.

```
1302 \newcommand*\ifglsused[3]{%
1303   \ifbool{glo@\glsdetoklabel{#1}@flag}{#2}{#3}%
1304 }
```

The following two commands will cause an error if the given condition fails:

`\glsdoifexists`

```
\glsdoifexists{<label>}{<code>}
```

Generate an error if entry specified by `<label>` doesn't exist, otherwise do `<code>`.

```
1305 \newcommand{\glsdoifexists}[2]{%
1306   \ifglsentryexists{#1}{#2}{%
1307     \PackageError{glossaries}{Glossary entry '\glsdetoklabel{#1}'
1308     has not been defined}{You need to define a glossary entry before you
1309     can use it.}}%
1310 }
```

`\glsdoifnoexists`

```
\glsdoifnoexists{<label>}{<code>}
```

The opposite: only do second argument if the entry doesn't exist. Generate an error message if it exists.

```
1311 \newcommand{\glsdoifnoexists}[2]{%
1312   \ifglsentryexists{#1}{%
```

```

1313 \PackageError{glossaries}{Glossary entry ‘\glsdetoklabel{#1}’ has already
1314 been defined}{-}{#2}%
1315 }

```

`\glsdoifexistsorwarn` `\glsdoifexistsorwarn{<label>}{<code>}`

Generate a warning if entry specified by *<label>* doesn't exist, otherwise do *<code>*.

```

1316 \newcommand{\glsdoifexistsorwarn}[2]{%
1317 \ifglsentryexists{#1}{#2}{%
1318 \GlossariesWarning{Glossary entry ‘\glsdetoklabel{#1}’
1319 has not been defined}%
1320 }%
1321 }

```

`\ifglshaschildren` `\ifglshaschildren{<label>}{<true part>}{<false part>}`

```

1322 \newcommand{\ifglshaschildren}[3]{%
1323 \glsdoifexists{#1}%
1324 {%
1325 \def\do@glshaschildren{#3}%
1326 \edef\@gls@thislabel{\glsdetoklabel{#1}}%
1327 \expandafter\for@gl@entries\expandafter
1328 [\csname glo@\@gls@thislabel @type\endcsname]
1329 {\glo@label}%
1330 {%
1331 \letcs@glo@parent{glo@\glo@label @parent}%
1332 \ifdefequal\@gls@thislabel\glo@parent
1333 {%
1334 \def\do@glshaschildren{#2}%
1335 \@endfortrue
1336 }%
1337 }%
1338 }%
1339 \do@glshaschildren
1340 }%
1341 }

```

`\ifglshasparent` `\ifglshasparent{<label>}{<true part>}{<false part>}`

```

1342 \newcommand{\ifglshasparent}[3]{%
1343 \glsdoifexists{#1}%
1344 {%
1345 \ifcsemtyp{glo@\glsdetoklabel{#1}@parent}{#3}{#2}%
1346 }%
1347 }

```

```

\ifglshasdesc \ifglshasdesc{<label>}{<true part>}{<false part>}
1348 \newcommand*\ifglshasdesc}[3]{%
1349 \ifcseempty{glo@glstdetoklabel{#1}@desc}%
1350 {#3}%
1351 {#2}%
1352 }

```

```

\ifglstdescsuppressed \ifglstdescsuppressed{<label>}{<true part>}{<false part>} Does <true part>
if the description is just \nopostdesc otherwise does <false part>.
1353 \newcommand*\ifglstdescsuppressed}[3]{%
1354 \ifcsequal{glo@glstdetoklabel{#1}@desc}{@no@post@desc}%
1355 {#2}%
1356 {#3}%
1357 }

```

```

\ifglshassymbol \ifglshassymbol{<label>}{<true part>}{<false part>}
1358 \newcommand*\ifglshassymbol}[3]{%
1359 \letcs{@glo@symbol}{glo@glstdetoklabel{#1}@symbol}%
1360 \ifdefempty@glo@symbol
1361 {#3}%
1362 {%
1363 \ifdefequal@glo@symbol@gls@default@value
1364 {#3}%
1365 {#2}%
1366 }%
1367 }

```

```

\ifglshaslong \ifglshaslong{<label>}{<true part>}{<false part>}
1368 \newcommand*\ifglshaslong}[3]{%
1369 \letcs{@glo@long}{glo@glstdetoklabel{#1}@long}%
1370 \ifdefempty@glo@long
1371 {#3}%
1372 {%
1373 \ifdefequal@glo@long@gls@default@value
1374 {#3}%
1375 {#2}%
1376 }%
1377 }

```

```

\ifglshasshort \ifglshasshort{<label>}{<true part>}{<false part>}
1378 \newcommand*\ifglshasshort}[3]{%
1379 \letcs{@glo@short}{glo@glstdetoklabel{#1}@short}%
1380 \ifdefempty@glo@short
1381 {#3}%
1382 {%
1383 \ifdefequal@glo@short@gls@default@value
1384 {#3}%
1385 {#2}%

```

```
1386 }%
1387 }
```

```
\ifglshasfield \ifglshasfield{<field>}{<label>}{<>true part>}{<>false part>}
```

```
1388 \newcommand*{\ifglshasfield}[4]{%
1389   \glsdoifexists{#2}%
1390   {%
1391     \letcs{\@glo@thisvalue}{glo@glsdetoklabel{#2}@#1}%
      First check supplied field label is defined.
1392     \ifdef\@glo@thisvalue
1393     {%
      Is defined, so now check if empty.
1394       \ifdefempty\@glo@thisvalue
1395       {%
      Is empty, so doesn't have field set.
1396         #4%
1397       }%
1398     }%
      Not empty, so check if set to \@gls@default@value
1399       \ifdequal\@glo@thisvalue\@gls@default@value{#4}{#3}%
1400     }%
1401   }%
1402 }%
      Field given isn't defined, so check if mapping exists.
1403   \@gls@fetchfield{\@gls@thisfield}{#1}%
      If \@gls@thisfield is defined, we've found a map. If not, the field supplied
      doesn't exist.
1404     \ifdef\@gls@thisfield
1405     {%
      Is defined, so now check if empty.
1406       \letcs{\@glo@thisvalue}{glo@glsdetoklabel{#2}\@gls@thisfield}%
1407       \ifdefempty\@glo@thisvalue
1408       {%
      Is empty so field hasn't been set.
1409         #4%
1410       }%
1411     }%
      Isn't empty so check if it's been set to \@gls@default@value.
1412       \ifdequal\@glo@thisvalue\@gls@default@value{#4}{#3}%
1413     }%
1414   }%
1415 }
```

Not defined.

```
1416     \GlossariesWarning{Unknown entry field ‘#1’}%
1417     #4%
1418     }%
1419 }%
1420 }%
1421 }
```

1.6 Defining new glossaries

A comma-separated list of glossary names is stored in `\@glo@types`. When a new glossary type is created, its identifying name is added to this list. This is used by commands that iterate through all glossaries (such as `\makeglossaries` and `\printglossaries`).

`\@glo@types`

```
1422 \newcommand*\@glo@types}{,}
```

`\provide@newglossary` If the user removes the glossary package from their document, ensure the next run doesn't throw a load of undefined control sequence errors when the aux file is parsed.

```
1423 \newcommand*\@gls@provide@newglossary{%
1424   \protected@write\@auxout{}\string\providecommand\string\@newglossary[4]{}%

```

Only need to do this once.

```
1425   \let\@gls@provide@newglossary\relax
1426 }
```

`\defglsentryfmt` Allow different glossaries to have different display styles.

```
1427 \newcommand*\defglsentryfmt}[2][\glsdefaulttype]{%
1428   \csgdef{gls@#1@entryfmt}{#2}%
1429 }
```

`\gls@doentryfmt`

```
1430 \newcommand*\gls@doentryfmt}[1]{\csuse{gls@#1@entryfmt}}
```

`\@gls@forbidtexext` As a security precaution, don't allow the user to specify a 'tex' extension for any of the glossary files. (Just in case a seriously confused novice user doesn't know what they're doing.) The argument must be a control sequence whose replacement text is the requested extension.

```
1431 \newcommand*\@gls@forbidtexext}[1]{%
1432   \ifboolexpr{test {\ifdefstring{#1}{tex}}
1433     or test {\ifdefstring{#1}{TEX}}}
1434   {%
1435     \def#1{nottex}%
1436     \PackageError{glossaries}%
1437       {Forbidden ‘.tex’ extension replaced with ‘.nottex’}%
1438     {I’m sorry, I can’t allow you to do something so reckless.\MessageBreak

```

```

1439     Don't use '.tex' as an extension for a temporary file.}%
1440 }%
1441 {%
1442 }%
1443 }

```

A new glossary type is defined using `\newglossary`. Syntax:

```

\newglossary[⟨log-ext⟩]{⟨name⟩}{⟨in-ext⟩}{⟨out-ext⟩}
{⟨title⟩}[⟨counter⟩]

```

where `⟨log-ext⟩` is the extension of the `makeindex` transcript file, `⟨in-ext⟩` is the extension of the glossary input file (read in by `\printglossary` and created by `makeindex`), `⟨out-ext⟩` is the extension of the glossary output file which is read in by `makeindex` (lines are written to this file by the `\glossary` command), `⟨title⟩` is the title of the glossary that is used in `\glossarysection` and `⟨counter⟩` is the default counter to be used by entries belonging to this glossary. The `makeglossaries` Perl script reads in the relevant extensions from the auxiliary file, and passes the appropriate file names and switches to `makeindex`.

`\newglossary`

```

1444 \newcommand*{\newglossary}{\@ifstar\s@newglossary\@ns@newglossary}

```

`\s@newglossary` The starred version will construct the extension based on the label.

```

1445 \newcommand*{\s@newglossary}[2]{%
1446 \ns@newglossary[#1-glg]{#1}{#1-gls}{#1-glo}{#2}%
1447 }

```

`\ns@newglossary` Define the unstarred version.

```

1448 \newcommand*{\ns@newglossary}[5][glg]{%
1449 \ifglossaryexists{#2}%
1450 {%
1451 \PackageError{glossaries}{Glossary type '#2' already exists}{%
1452 You can't define a new glossary called '#2' because it already
1453 exists}%
1454 }%
1455 {%

```

Check if default has been set

```

1456 \ifundef\glsdefaulttype
1457 {%
1458 \gdef\glsdefaulttype{#2}%
1459 }{%

```

Add this to the list of glossary types:

```

1460 \toks@{#2}\edef\@glo@types{\@glo@types\the\toks@,}%

```

Define a comma-separated list of labels for this glossary type, so that all the entries for this glossary can be reset with a single command. When a new entry is created, its label is added to this list.

```
1461 \expandafter\gdef\csname glolist@#2\endcsname{,}%
```

Store the file extensions:

```
1462 \expandafter\edef\csname @glotype@#2@log\endcsname{#1}%
1463 \expandafter\edef\csname @glotype@#2@in\endcsname{#3}%
1464 \expandafter\edef\csname @glotype@#2@out\endcsname{#4}%
1465 \expandafter\@gls@forbidtextext\csname @glotype@#2@log\endcsname
1466 \expandafter\@gls@forbidtextext\csname @glotype@#2@in\endcsname
1467 \expandafter\@gls@forbidtextext\csname @glotype@#2@out\endcsname
```

Store the title:

```
1468 \expandafter\def\csname @glotype@#2@title\endcsname{#5}%
```

```
1469 \@gls@provide@newglossary
```

```
1470 \protected@write\@auxout{}{\string\@newglossary{#2}{#1}{#3}{#4}}%
```

How to display this entry in the document text (uses `\glsentry` by default).

This can be redefined by the user later if required (see `\defglsentry`). This may already have been defined if this has been specified as a list of acronyms.

```
1471 \ifcsundef{gls@#2@entryfmt}%
1472 {%
1473   \defglsentryfmt[#2]{\glsentryfmt}%
1474 }%
1475 {}%
```

Define sort counter if required:

```
1476 \@gls@defsortcount{#2}%
```

Find out if the final optional argument has been specified, and use it to set the counter associated with this glossary. (Uses `\glscounter` if no optional argument is present.)

```
1477 \@ifnextchar[{\@gls@setcounter{#2}}%
1478   {\@gls@setcounter{#2}[\glscounter]}%
1479 }
```

`\altnewglossary`

```
1480 \newcommand*{\altnewglossary}[3]{%
1481   \newglossary[#2-glg]{#1}{#2-gls}{#2-glo}{#3}%
1482 }
```

Only define new glossaries in the preamble:

```
1483 \@onlypreamble{\newglossary}
```

Only define new glossaries before `\makeglossaries`

```
1484 \@onlypremakeg\newglossary
```

`\@newglossary` is used to specify the file extensions for the `makeindex` input, output and transcript files. It is written to the auxiliary file by `\newglossary`. Since it is not used by \LaTeX , `\@newglossary` simply ignores its arguments.

`\@newglossary`

```
1485 \newcommand*{\@newglossary}[4]{} 
```

Store counter to be used for given glossary type (the first argument is the glossary label, the second argument is the name of the counter):

`\@gls@setcounter`

```
1486 \def\@gls@setcounter#1[#2]{% 
```

```
1487   \expandafter\def\csname @gls@#1@counter\endcsname{#2}% 
```

Add counter to xindy list, if not already added:

```
1488   \ifglxindy 
```

```
1489     \GlsAddXdyCounters{#2}% 
```

```
1490   \fi 
```

```
1491 }
```

Get counter associated with given glossary (the argument is the glossary label):

`\@gls@getcounter`

```
1492 \newcommand*{\@gls@getcounter}[1]{% 
```

```
1493   \csname @gls@#1@counter\endcsname 
```

```
1494 }
```

Define the main glossary. This will be the first glossary to be displayed when using `\printglossaries`.

```
1495 \glsdefmain 
```

Define the “acronym” glossaries if required.

```
1496 \@gls@do@acronymsdef 
```

Define the “symbols”, “numbers” and “index” glossaries if required.

```
1497 \@gls@do@symbolsdef 
```

```
1498 \@gls@do@numbersdef 
```

```
1499 \@gls@do@indexdef 
```

`\newignoredglossary` Creates a new glossary that doesn't have associated files. This glossary is ignored by and commands that iterate over glossaries, such as `\printglossaries`, and won't work with commands like `\printglossary`. It's intended for entries that are so commonly-known they don't require a glossary.

```
1500 \newcommand*{\newignoredglossary}[1]{% 
```

```
1501   \ifdefempty\@ignored@glossaries 
```

```
1502   {% 
```

```
1503     \edef\@ignored@glossaries{#1}% 
```

```
1504   }% 
```

```
1505   {% 
```

```
1506     \eappto\@ignored@glossaries{,#1}% 
```

```
1507   }% 
```

```
1508   \csgdef{glolist@#1}{,}% 
```

```
1509   \ifcsundef{gls@#1@entryfmt}% 
```

```
1510   {% 
```

```

1511 \defglentryfmt[#1]{\glentryfmt}%
1512 }%
1513 {}%
1514 \ifdefempty\@gls@nohyperlist
1515 {%
1516 \renewcommand*\@gls@nohyperlist}{#1}%
1517 }%
1518 {}%
1519 \eappto\@gls@nohyperlist{,#1}%
1520 }%
1521 }

```

`@ignored@glossaries` List of ignored glossaries.

```

1522 \newcommand*\@ignored@glossaries{}

```

`\ifignoredglossary` Tests if the given glossary is an ignored glossary. Expansion is used in case the first argument is a control sequence.

```

1523 \newcommand*\ifignoredglossary}[3]{%
1524 \edef\@gls@igtype{#1}%
1525 \expandafter\DTLifinlist\expandafter
1526 {\@gls@igtype}{\@ignored@glossaries}{#2}{#3}%
1527 }

```

1.7 Defining new entries

New glossary entries are defined using `\newglossaryentry`. This command requires a label and a key-value list that defines the relevant information for that entry. The definition for these keys follows. Note that the name, description and symbol keys will be sanitized later, depending on the value of the package option `sanitize` (this means that if some of the keys haven't been defined, they can be constructed from the name and description key before they are sanitized).

name The name key indicates the name of the term being defined. This is how the term will appear in the glossary. The name key is required when defining a new glossary entry.

```

1528 \define@key{glossentry}{name}{%
1529 \def\@glo@name{#1}%
1530 }

```

description The description key is usually only used in the glossary, but can be made to appear in the text by redefining `\glentryfmt` or using `\defglentryfmt`. The description key is required when defining a new glossary entry. If a long description is required, use `\longnewglossaryentry` instead of `\newglossaryentry`.

```

1531 \define@key{glossentry}{description}{%
1532 \def\@glo@desc{#1}%
1533 }

```

descriptionplural

```
1534 \define@key{glossentry}{descriptionplural}{%
1535 \def\@glo@descplural{#1}%
1536 }
```

sort The sort key needs to be sanitized here (the sort key is provided for `makeindex`'s benefit, not for use in the document). The sort key is optional when defining a new glossary entry. If omitted, the value is given by $\langle name \rangle \langle description \rangle$.

```
1537 \define@key{glossentry}{sort}{%
1538 \def\@glo@sort{#1}}
```

text The text key determines how the term should appear when used in the document (i.e. outside of the glossary). If omitted, the value of the name key is used instead.

```
1539 \define@key{glossentry}{text}{%
1540 \def\@glo@text{#1}%
1541 }
```

plural The plural key determines how the plural form of the term should be displayed in the document. If omitted, the plural is constructed by appending `\glspluralsuffix` to the value of the text key.

```
1542 \define@key{glossentry}{plural}{%
1543 \def\@glo@plural{#1}%
1544 }
```

first The first key determines how the entry should be displayed in the document when it is first used. If omitted, it is taken to be the same as the value of the text key.

```
1545 \define@key{glossentry}{first}{%
1546 \def\@glo@first{#1}%
1547 }
```

firstplural The `firstplural` key is used to set the plural form for first use, in the event that the plural is required the first time the term is used. If omitted, it is constructed by appending `\glspluralsuffix` to the value of the first key.

```
1548 \define@key{glossentry}{firstplural}{%
1549 \def\@glo@firstplural{#1}%
1550 }
```

`\@gls@default@value`

```
1551 \newcommand*{\@gls@default@value}{\relax}
```

symbol The symbol key is ignored by most of the predefined glossary styles, and defaults to `\relax` if omitted. It is provided for glossary styles that require an associated symbol, as well as a name and description. To make this value appear in the glossary, you need to redefine `\glossentry`. If you want this value to

appear in the text when the term is used by commands like `\gls`, you will need to change `\glsentryfmt` (or use for `\defglsentryfmt` individual glossaries).

```
1552 \define@key{glossentry}{symbol}{%
1553 \def\@glo@symbol{#1}%
1554 }
```

`symbolplural`

```
1555 \define@key{glossentry}{symbolplural}{%
1556 \def\@glo@symbolplural{#1}%
1557 }
```

`type` The `type` key specifies to which glossary this entry belongs. If omitted, the default glossary is used.

```
1558 \define@key{glossentry}{type}{%
1559 \def\@glo@type{#1}}
```

`counter` The `counter` key specifies the name of the counter associated with this glossary entry:

```
1560 \define@key{glossentry}{counter}{%
1561 \ifcsundef{c@#1}%
1562 {%
1563 \PackageError{glossaries}%
1564 {There is no counter called ‘#1’}%
1565 {%
1566 The counter key should have the name of a valid counter
1567 as its value%
1568 }%
1569 }%
1570 {%
1571 \def\@glo@counter{#1}%
1572 }%
1573 }
```

`see` The `see` key specifies a list of cross-references

```
1574 \define@key{glossentry}{see}{%
1575 \gls@checkseeallowed
1576 \def\@glo@see{#1}%
1577 \@glo@seeautonumberlist
1578 }
```

`gls@checkseeallowed`

```
1579 \newcommand*{\gls@checkseeallowed}{%
1580 \PackageError{glossaries}%
1581 {‘see’ key may only be used after \string\makeglossaries\space
1582 or \string\makenoidxglossaries}%
1583 {You must use \string\makeglossaries\space
1584 or \string\makenoidxglossaries\space before defining
1585 any entries that have a ‘see’ key}%
1586 }
```

parent The parent key specifies the parent entry, if required.

```
1587 \define@key{glossentry}{parent}{%
1588 \def\@glo@parent{#1}}
```

nonumberlist The nonumberlist key suppresses or activates the number list for the given entry.

```
1589 \define@choicekey{glossentry}{nonumberlist}[\val\nr]{true,false}[true]{%
1590 \ifcase\nr\relax
1591 \def\@glo@prefix{\glsnonextpages}%
1592 \else
1593 \def\@glo@prefix{\glsnextpages}%
1594 \fi
1595 }
```

Define some generic user keys. (Additional keys can be added by the user.)

user1

```
1596 \define@key{glossentry}{user1}{%
1597 \def\@glo@useri{#1}%
1598 }
```

user2

```
1599 \define@key{glossentry}{user2}{%
1600 \def\@glo@userii{#1}%
1601 }
```

user3

```
1602 \define@key{glossentry}{user3}{%
1603 \def\@glo@useriii{#1}%
1604 }
```

user4

```
1605 \define@key{glossentry}{user4}{%
1606 \def\@glo@useriv{#1}%
1607 }
```

user5

```
1608 \define@key{glossentry}{user5}{%
1609 \def\@glo@userv{#1}%
1610 }
```

user6

```
1611 \define@key{glossentry}{user6}{%
1612 \def\@glo@uservi{#1}%
1613 }
```

`short` This key is provided for use by `\newacronym`. It's not designed for general purpose use, so isn't described in the user manual.

```
1614 \define@key{glossentry}{short}{%
1615   \def\@glo@short{#1}%
1616 }
```

`shortplural` This key is provided for use by `\newacronym`.

```
1617 \define@key{glossentry}{shortplural}{%
1618   \def\@glo@shortpl{#1}%
1619 }
```

`long` This key is provided for use by `\newacronym`.

```
1620 \define@key{glossentry}{long}{%
1621   \def\@glo@long{#1}%
1622 }
```

`longplural` This key is provided for use by `\newacronym`.

```
1623 \define@key{glossentry}{longplural}{%
1624   \def\@glo@longpl{#1}%
1625 }
```

`\@glsnoname` Define command to generate error if name key is missing.

```
1626 \newcommand*\@glsnoname{%
1627   \PackageError{glossaries}{name key required in
1628   \string\newglossaryentry\space for entry '\@glo@label'}{You
1629   haven't specified the entry name}}
```

`\@glsnodesc` Define command to generate error if description key is missing.

```
1630 \newcommand*\@glsnodesc{%
1631   \PackageError{glossaries}
1632   {%
1633     description key required in \string\newglossaryentry\space
1634     for entry '\@glo@label'%
1635   }%
1636   {%
1637     You haven't specified the entry description%
1638   }%
1639 }
```

`\@glsdefaultplural` Now obsolete. Don't use.

```
1640 \newcommand*\@glsdefaultplural{}
```

`\@gls@missingnumberlist` Define a command to generate warning when numberlist not set.

```
1641 \newcommand*\@gls@missingnumberlist}[1]{%
1642   ??%
1643   \ifglssavenumberlist
1644     \GlossariesWarning{Missing number list for entry '#1'.
1645     Maybe makeglossaries + rerun required.}%
1646 }
```

```

1646 \else
1647   \PackageError{glossaries}%
1648   {Package option ‘savenumberlist=true’ required.}%
1649   {%
1650     You must use the ‘savenumberlist’ package option
1651     to reference location lists.%
1652   }%
1653 \fi
1654 }

```

`\@glsdefaultsort` Define command to set default sort.

```
1655 \newcommand*{\@glsdefaultsort}{\@glo@name}
```

`\gls@level` Register to increment entry levels.

```
1656 \newcount\gls@level
```

`@gls@noexpand@field`

```

1657 \newcommand{\@gls@noexpand@field}[3]{%
1658 \expandafter\global\expandafter
1659 \let\csname glo@#1@#2\endcsname#3%
1660 }

```

`gls@noexpand@fields`

```

1661 \newcommand{\@gls@noexpand@fields}[4]{%
1662 \ifcsdef{gls@assign@#3@field}
1663 {%
1664 \ifdefequal{#4}{\@gls@default@value}%
1665 {%
1666 \edef\@gls@value{\expandonce{#1}}%
1667 \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1668 }%
1669 }%
1670 \csuse{gls@assign@#3@field}{#2}{#4}%
1671 }%
1672 }%
1673 {%
1674 \ifdefequal{#4}{\@gls@default@value}%
1675 {%
1676 \edef\@gls@value{\expandonce{#1}}%
1677 \@gls@noexpand@field{#2}{#3}{\@gls@value}%
1678 }%
1679 }%
1680 \@gls@noexpand@field{#2}{#3}{#4}%
1681 }%
1682 }%
1683 }

```

`\@@gls@expand@field`

```

1684 \newcommand{\@gls@expand@field}[3]{%
1685 \expandafter
1686 \protected@xdef\csname glo@#1@#2\endcsname{#3}%
1687 }

```

@gls@expand@fields

```

1688 \newcommand{\@gls@expand@fields}[4]{%
1689 \ifcsdef{gls@assign@#3@field}
1690 {%
1691 \ifdefequal{#4}{\@gls@default@value}%
1692 {%
1693 \edef\@gls@value{\expandonce{#1}}%
1694 \csuse{gls@assign@#3@field}{#2}{\@gls@value}%
1695 }%
1696 {%
1697 \expandafter\@gls@startswithexpandonce#4\relax\relax@gls@endcheck
1698 {%
1699 \@@gls@expand@field{#2}{#3}{#4}%
1700 }%
1701 {%
1702 \csuse{gls@assign@#3@field}{#2}{#4}%
1703 }%
1704 }%
1705 }%
1706 {%
1707 \ifdefequal{#4}{\@gls@default@value}%
1708 {%
1709 \@@gls@expand@field{#2}{#3}{#1}%
1710 }%
1711 {%
1712 \@@gls@expand@field{#2}{#3}{#4}%
1713 }%
1714 }%
1715 }

```

startswithexpandonce

```

1716 \def\@gls@expandonce{\expandonce}
1717 \def\@gls@startswithexpandonce#1#2@gls@endcheck#3#4{%
1718 \def\@gls@tmp{#1}%
1719 \ifdefequal{\@gls@expandonce}{\@gls@tmp}{#3}{#4}%
1720 }

```

\gls@assign@field `\gls@assign@field{<def value>}{<glossary type>}{<field>}{<tmp cs>}`

Assigns an entry field. Expansion performed by default (except for name, symbol and description where backward compatibility required). If *<tmp cs>* is *<@gls@default@value>*, *<def value>* is used instead.

```
1721 \let@gls@assign@field\@gls@expand@fields
```



```

\glsexpandfields Fully expand values when assigning fields (except for specific fields that are
                  overridden by \glssetnoexpandfield).
1722 \newcommand*\glsexpandfields{%
1723   \let\gls@assign@field\@gls@expand@fields
1724 }

\glsnoexpandfields Don't expand values when assigning fields (except for specific fields that are
                   overridden by \glssetexpandfield).
1725 \newcommand*\glsnoexpandfields{%
1726   \let\gls@assign@field\@gls@noexpand@fields
1727 }

\newglossaryentry Define \newglossaryentry {<label>} {<key-val list>}. There are two required
                  fields in <key-val list>: name (or parent) and description. (See above.)
1728 \newrobustcmd{\newglossaryentry}[2]{%
    Check to see if this glossary entry has already been defined:
1729   \glsdoifnoexists{#1}%
1730   {%
1731     \gls@defglossaryentry{#1}{#2}%
1732   }%
1733 }

\provideglossaryentry Like \newglossaryentry but does nothing if the entry has already been de-
                    fined.
1734 \newrobustcmd{\provideglossaryentry}[2]{%
1735   \ifglsentryexists{#1}%
1736   }%
1737   {%
1738     \gls@defglossaryentry{#1}{#2}%
1739   }%
1740 }
1741 \@onlypreamble{\provideglossaryentry}

\new@glossaryentry For use in document environment.
1742 \newrobustcmd{\new@glossaryentry}[2]{%
1743   \ifundef\@gls@deffile
1744   {%
1745     \global\newwrite\@gls@deffile
1746     \immediate\openout\@gls@deffile=\jobname.glsdefs
1747   }%
1748   {%
1749     \ifglsentryexists{#1}{}%
1750     {%
1751       \gls@defglossaryentry{#1}{#2}%
1752     }%
1753     \@gls@writedef{#1}%
1754 }

```

```

1755 \AtBeginDocument
1756 {
1757   \makeatletter
1758   \InputIfFileExists{\jobname.glsdefs}{\}{}%
1759   \makeatother
1760   \let\newglossaryentry\new@glossaryentry
1761 }
1762 \AtEndDocument{\ifdef\@gls@deffile{\closeout\@gls@deffile}{}}

```

`\@gls@writedef` Writes glossary entry definition to `\@gls@deffile`.

```

1763 \newcommand*\@gls@writedef}[1]{%
1764   \immediate\write\@gls@deffile
1765   {%
1766     \string\ifglstentryexists{#1}{}\glspercentchar^^J%
1767     \expandafter\@gobble\string\{\glspercentchar^^J%
1768     \string\gls@defglossaryentry{\glsdetoklabel{#1}}\glspercentchar^^J%
1769     \expandafter\@gobble\string\{\glspercentchar%
1770   }%

```

Write key value information:

```

1771   \@for\@gls@map:=\@gls@keymap\do
1772   {%
1773     \edef\glo@value{\expandafter\expandonce
1774       \csname glo@\glsdetoklabel{#1}\@expandafter
1775       \@secondoftwo\@gls@map\endcsname}%
1776     \@onelevel@sanitize\glo@value
1777     \immediate\write\@gls@deffile
1778     {%
1779       \expandafter\@firstoftwo\@gls@map
1780       =\expandafter\@gobble\string\{\glo@value\expandafter\@gobble\string\},%
1781       \glspercentchar%
1782     }%
1783   }%

```

Provide hook:

```

1784   \gls.writedefhook
1785   \immediate\write\@gls@deffile
1786   {%
1787     \glspercentchar^^J%
1788     \expandafter\@gobble\string\}\glspercentchar^^J%
1789     \expandafter\@gobble\string\}\glspercentchar%
1790   }%
1791 }

```

`\@gls@keymap` List of entry definition key names and corresponding tag in control sequence used to store the value.

```

1792 \newcommand*\@gls@keymap){%
1793   {name}{name},%
1794   {sort}{sortvalue},% unescaped sort value
1795   {type}{type},%

```

```

1796 {first}{first},%
1797 {firstplural}{firstpl},%
1798 {text}{text},%
1799 {plural}{plural},%
1800 {description}{desc},%
1801 {descriptionplural}{descplural},%
1802 {symbol}{symbol},%
1803 {symbolplural}{symbolplural},%
1804 {user1}{useri},%
1805 {user2}{userii},%
1806 {user3}{useriii},%
1807 {user4}{useriv},%
1808 {user5}{userv},%
1809 {user6}{uservi},%
1810 {long}{long},%
1811 {longplural}{longpl},%
1812 {short}{short},%
1813 {shortplural}{shortpl},%
1814 {counter}{counter},%
1815 {parent}{parent}%
1816 }

```

```
\@gls@fetchfield \@gls@fetchfield{<cs>}{<field>}
```

Fetches the internal field label from the given user *<field>* and stores in *<cs>*.

```
1817 \newcommand*\@gls@fetchfield}[2]{%
```

Ensure user field name is fully expanded

```
1818 \edef\@gls@thisval{#2}%
```

Iterate through known mappings until we find the one for this field.

```

1819 \@for\@gls@map:=\@gls@keymap\do{%
1820 \edef\@this@key{\expandafter\@firstoftwo\@gls@map}%
1821 \ifdefequal{\@this@key}{\@gls@thisval}%
1822 {%
```

Found it.

```
1823 \edef#1{\expandafter\@secondoftwo\@gls@map}%
```

Break out of loop.

```

1824 \@endfortrue
1825 }%
1826 {}%
1827 }%
1828 }

```

```
\glsaddkey \glsaddkey{<key>}{<default value>}{<no link cs>}{<no link ucfirst cs>}{<link cs>}{<link ucfirst cs>}{<link allcaps cs>}
```

Allow user to add their own custom keys.

```
1829 \newcommand*{\glsaddkey}{\@ifstar\sglsaddkey\@glsaddkey}
```

Starred version switches on expansion for this key.

```
1830 \newcommand*{\@glsaddkey}[1]{%
1831   \key@ifundefined{glossentry}{#1}%
1832   {%
1833     \expandafter\newcommand\expandafter*\expandafter
1834     {\csname gls@assign@#1@field\endcsname}[2]{%
1835       \@gls@expand@field{##1}{#1}{##2}%
1836     }%
1837   }%
1838   }%
1839   \@glsaddkey{#1}%
1840 }
```

Unstarred version doesn't override default expansion.

```
1841 \newcommand*{\@glsaddkey}[7]{%
```

Check the specified key doesn't already exist.

```
1842   \key@ifundefined{glossentry}{#1}%
1843   {%
```

Set up the key.

```
1844     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
1845     \appto\@gls@keymap{, #1}{#1}}%
```

Set the default value.

```
1846     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
```

Assignment code.

```
1847     \appto\@newglossaryentryposthook{%
1848       \letcs{\@glo@tmp}{@glo@#1}%
1849       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
1850     }%
```

Define the no-link commands.

```
1851     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
1852     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%
```

Now for the commands with links. First the version with no case change:

```
1853     \ifcsdef{@gls@user@#1@}%
1854     {%
1855       \PackageError{glossaries}%
1856       {Can't define '\string#5' as helper command
1857       '\expandafter\string\csname @gls@user@#1@\endcsname' already exists}%
1858       }%
1859     }%
1860     {%
1861       \expandafter\newcommand\expandafter*\expandafter
1862       {\csname @gls@user@#1\endcsname}[2][ ]{%
1863         \new@ifnextchar[%
```

```

1864         {\csuse{@gls@user@#1@}{##1}{##2}}%
1865         {\csuse{@gls@user@#1@}{##1}{##2} []}}%
1866     \csdef{@gls@user@#1@}##1##2[##3]{%
1867         \@gls@field@link{##1}{##2}{#3{##2}##3}%
1868     }%
1869     \newrobustcmd*{#5}{%
1870         \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
1871     }%

```

Next the version with the first letter converted to upper case:

```

1872     \ifcsdef{@Gls@user@#1@}%
1873     {%
1874         \PackageError{glossaries}%
1875         {Can't define '\string#6' as helper command
1876         '\expandafter\string\csname @Gls@user@#1@\endcsname' already exists}%
1877     }%
1878     }%
1879     {%
1880         \expandafter\newcommand\expandafter*\expandafter
1881         {\csname @Gls@user@#1\endcsname}[2] []{%
1882             \new@ifnextchar[%
1883                 {\csuse{@Gls@user@#1@}{##1}{##2}}%
1884                 {\csuse{@Gls@user@#1@}{##1}{##2} []}}%
1885         \csdef{@Gls@user@#1@}##1##2[##3]{%
1886             \@gls@field@link{##1}{##2}{#4{##2}##3}%
1887         }%
1888         \newrobustcmd*{#6}{%
1889             \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
1890         }%

```

Finally the all caps version:

```

1891     \ifcsdef{@GLS@user@#1@}%
1892     {%
1893         \PackageError{glossaries}%
1894         {Can't define '\string#7' as helper command
1895         '\expandafter\string\csname @GLS@user@#1@\endcsname' already exists}%
1896     }%
1897     }%
1898     {%
1899         \expandafter\newcommand\expandafter*\expandafter
1900         {\csname @GLS@user@#1\endcsname}[2] []{%
1901             \new@ifnextchar[%
1902                 {\csuse{@GLS@user@#1@}{##1}{##2}}%
1903                 {\csuse{@GLS@user@#1@}{##1}{##2} []}}%
1904         \csdef{@GLS@user@#1@}##1##2[##3]{%
1905             \@gls@field@link{##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
1906         }%
1907         \newrobustcmd*{#7}{%

```

```

1908     \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
1909   }%
1910 }%
1911 {%
1912   \PackageError{glossaries}{Key ‘#1’ already exists}{}%
1913 }%
1914 }

```

`\gls>writedefhook`

```
1915 \newcommand*\gls>writedefhook{}
```

`\gls@assign@desc`

```

1916 \newcommand*\gls@assign@desc}[1]{%
1917   \gls@assign@field{#1}{desc}{\@glo@desc}%
1918   \gls@assign@field{\@glo@desc}{#1}{descplural}{\@glo@descplural}%
1919 }

```

`\longnewglossaryentry`

```

1920 \newcommand{\longnewglossaryentry}[3]{%
1921   \glsdoifnoexists{#1}%
1922   {%
1923     \bgroup
1924     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1925     \long\def\@newglossaryentryprehook{%
1926       \long\def\@glo@desc{#3\leavevmode\unskip\nopostdesc}%
1927       \@org@newglossaryentryprehook
1928     }%
1929     \renewcommand*\gls@assign@desc}[1]{%
1930       \global\cslet{glo@glsdetoklabel{#1}@desc}{\@glo@desc}%
1931       \global\cslet{glo@glsdetoklabel{#1}@descplural}{\@glo@desc}%
1932     }
1933     \gls@defglossaryentry{#1}{#2}%
1934   \egroup
1935 }
1936 }

```

Only allowed in the preamble. (Otherwise a long description could cause problems when writing the entry definition to the temporary file.)

```
1937 \@onlypreamble{\longnewglossaryentry}
```

`\provideglossaryentry` As the above but only defines the entry if it doesn't already exist.

```

1938 \newcommand{\longprovideglossaryentry}[3]{%
1939   \ifglsentryexists{#1}{}%
1940   {\longnewglossaryentry{#1}{#2}{#3}}%
1941 }
1942 \@onlypreamble{\longprovideglossaryentry}

```

`\gls@defglossaryentry`

```
\gls@defglossaryentry{<label>}{<key-val list>}
```

Defines a new entry without checking if it already exists.

```

1943 \newcommand{\gls@defglossaryentry}[2]{%
  Store label
1944   \edef\@glo@label{\glsdetoklabel{#1}}%
  Provide a means for user defined keys to reference the label:
1945   \let\glslabel\@glo@label
  Set up defaults. If the name or description keys are omitted, an error will be
  generated.
1946   \let\@glo@name\@glsnname
1947   \let\@glo@desc\@glsnodesc
1948   \let\@glo@descplural\@gls@default@value
1949   \let\@glo@type\@gls@default@value
1950   \let\@glo@symbol\@gls@default@value
1951   \let\@glo@symbolplural\@gls@default@value
1952   \let\@glo@text\@gls@default@value
1953   \let\@glo@plural\@gls@default@value
  Using \let instead of \def to make later comparison avoid expansion issues.
  (Thanks to Ulrich Diez for suggesting this.)
1954   \let\@glo@first\@gls@default@value
1955   \let\@glo@firstplural\@gls@default@value
  Set the default sort:
1956   \let\@glo@sort\@gls@default@value
  Set the default counter:
1957   \let\@glo@counter\@gls@default@value
1958   \def\@glo@see{}%
1959   \def\@glo@parent{}%
1960   \def\@glo@prefix{}%
1961   \def\@glo@useri{}%
1962   \def\@glo@userii{}%
1963   \def\@glo@useriii{}%
1964   \def\@glo@useriv{}%
1965   \def\@glo@userv{}%
1966   \def\@glo@uservi{}%
1967   \def\@glo@short{}%
1968   \def\@glo@shortpl{}%
1969   \def\@glo@long{}%
1970   \def\@glo@longpl{}%
```

Add start hook in case another package wants to add extra keys.

```
1971 \@newglossaryentryprehook
```

Extract key-val information from third parameter:

```
1972 \setkeys{glossentry}{#2}%
```

Check there is a default glossary.

```
1973 \ifundef\glsdefaulttype
```

```
1974 {%
```

```
1975 \PackageError{glossaries}%
```

```
1976 {No default glossary type (have you used 'nomain')}%
```

```
1977 {If you use package option 'nomain' you must define
```

```
1978 a new glossary before you can define entries}%
```

```
1979 }%
```

```
1980 {}%
```

Assign type. This must be fully expandable

```
1981 \gls@assign@field{\glsdefaulttype}{\@glo@label}{type}{\@glo@type}%
```

```
1982 \edef\@glo@type{\glsentrytype{\@glo@label}}%
```

Check to see if this glossary type has been defined, if it has, add this label to the relevant list, otherwise generate an error.

```
1983 \ifcsundef{glolist@\@glo@type}%
```

```
1984 {%
```

```
1985 \PackageError{glossaries}%
```

```
1986 {Glossary type '\@glo@type' has not been defined}%
```

```
1987 {You need to define a new glossary type, before making entries
```

```
1988 in it}%
```

```
1989 }%
```

```
1990 {}%
```

Check if it's an ignored glossary

```
1991 \ifignoredglossary\@glo@type
```

```
1992 {%
```

The description may be omitted for an entry in an ignored glossary.

```
1993 \ifx\@glo@desc\@glsnodesc
```

```
1994 \let\@glo@desc\@empty
```

```
1995 \fi
```

```
1996 }%
```

```
1997 {%
```

```
1998 }%
```

```
1999 \protected@edef\@glolist@{\csname glolist@\@glo@type\endcsname}%
```

```
2000 \expandafter\xdef\csname glolist@\@glo@type\endcsname{%
```

```
2001 \@glolist@{\@glo@label},}%
```

```
2002 }%
```

Initialise level to 0.

```
2003 \gls@level=0\relax
```

Has this entry been assigned a parent?

```
2004 \ifx\@glo@parent\@empty
```


Doesn't have a parent. Set `\glo@<label>@parent` to empty.

```
2005     \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
2006     \else
```

Has a parent. Check to ensure this entry isn't its own parent.

```
2007     \ifdefequal\@glo@label\@glo@parent%
2008     {%
2009     \PackageError{glossaries}{Entry '@@glo@label' can't be its own parent}{}%
2010     \def\@glo@parent{}%
2011     \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
2012     }%
2013     {%
```

Check the parent exists:

```
2014     \ifglentryexists{\@glo@parent}%
2015     {%
```

Parent exists. Set `\glo@<label>@parent`.

```
2016     \expandafter\xdef\csname glo@\@glo@label @parent\endcsname{%
2017     \@glo@parent}%
```

Determine level.

```
2018     \gls@level=\csname glo@\@glo@parent @level\endcsname\relax
2019     \advance\gls@level by 1\relax
```

If name hasn't been specified, use same as the parent name

```
2020     \ifx\@glo@name\@gls@name
2021     \expandafter\let\expandafter\@glo@name
2022     \csname glo@\@glo@parent @name\endcsname
```

If name and plural haven't been specified, use same as the parent

```
2023     \ifx\@glo@plural\@gls@default@value
2024     \expandafter\let\expandafter\@glo@plural
2025     \csname glo@\@glo@parent @plural\endcsname
2026     \fi
2027     \fi
2028     }%
2029     {%
```

Parent doesn't exist, so issue an error message and change this entry to have no parent

```
2030     \PackageError{glossaries}%
2031     {%
2032     Invalid parent '@@glo@parent'
2033     for entry '@@glo@label' - parent doesn't exist%
2034     }%
2035     {%
2036     Parent entries must be defined before their children%
2037     }%
2038     \def\@glo@parent{}%
2039     \expandafter\gdef\csname glo@\@glo@label @parent\endcsname{}%
2040     }%
```

```

2041     }%
2042   \fi

Set the level for this entry
2043   \expandafter\xdef\csname glo@\@glo@label @level\endcsname{\number\gls@level}%

Define commands associated with this entry:
2044   \gls@assign@field{\@glo@name}{\@glo@label}{sortvalue}{\@glo@sort}%
2045   \letcs\@glo@sort{glo@\@glo@label @sortvalue}%
2046   \gls@assign@field{\@glo@name}{\@glo@label}{text}{\@glo@text}%
2047   \expandafter\gls@assign@field\expandafter
2048     {\csname glo@\@glo@label @text\endcsname\glspluralsuffix}%
2049     {\@glo@label}{plural}{\@glo@plural}%
2050   \expandafter\gls@assign@field\expandafter
2051     {\csname glo@\@glo@label @text\endcsname}%
2052     {\@glo@label}{first}{\@glo@first}%

If first has been specified, make the default by appending \glspluralsuffix,
otherwise make the default the value of the plural key.
2053   \ifx\@glo@first\@gls@default@value
2054     \expandafter\gls@assign@field\expandafter
2055       {\csname glo@\@glo@label @plural\endcsname}%
2056       {\@glo@label}{firstpl}{\@glo@firstplural}%
2057   \else
2058     \expandafter\gls@assign@field\expandafter
2059       {\csname glo@\@glo@label @first\endcsname\glspluralsuffix}%
2060       {\@glo@label}{firstpl}{\@glo@firstplural}%
2061   \fi

2062   \ifcsundef{@glo@type@\@glo@type @counter}%
2063   {%
2064     \def\@glo@defaultcounter{\glscounter}%
2065   }%
2066   {%
2067     \letcs\@glo@defaultcounter{@glo@type@\@glo@type @counter}%
2068   }%
2069   \gls@assign@field{\@glo@defaultcounter}{\@glo@label}{counter}{\@glo@counter}%
2070   \gls@assign@field{}{\@glo@label}{useri}{\@glo@useri}%
2071   \gls@assign@field{}{\@glo@label}{userii}{\@glo@userii}%
2072   \gls@assign@field{}{\@glo@label}{useriii}{\@glo@useriii}%
2073   \gls@assign@field{}{\@glo@label}{useriv}{\@glo@useriv}%
2074   \gls@assign@field{}{\@glo@label}{userv}{\@glo@userv}%
2075   \gls@assign@field{}{\@glo@label}{uservi}{\@glo@uservi}%
2076   \gls@assign@field{}{\@glo@label}{short}{\@glo@short}%
2077   \gls@assign@field{}{\@glo@label}{shortpl}{\@glo@shortpl}%
2078   \gls@assign@field{}{\@glo@label}{long}{\@glo@long}%
2079   \gls@assign@field{}{\@glo@label}{longpl}{\@glo@longpl}%
2080   \ifx\@glo@name\@glsnoname
2081     \@glsnoname
2082     \let\@gloname\@gls@default@value
2083   \fi

```

```
2084 \gls@assign@field{\@glo@label}{name}{\@glo@name}%
```

Set default numberlist if not defined:

```
2085 \ifcsundef{glo@\@glo@label @numberlist}%
2086 {%
2087 \csxdef{glo@\@glo@label @numberlist}{%
2088 \noexpand@gls@missingnumberlist{\@glo@label}}%
2089 }%
2090 {}%
```

The smaller and smallcaps options set the description to \@glo@first. Need to check for this, otherwise it won't get expanded if the description gets sanitized.

```
2091 \def@glo@@desc{\@glo@first}%
2092 \ifx@glo@desc@glo@@desc
2093 \let@glo@desc@glo@first
2094 \fi
2095 \ifx@glo@desc@glsnodesc
2096 \let@glo@desc@glsnodesc
2097 \let@glo@desc@gls@default@value
2098 \fi
2099 \gls@assign@desc{\@glo@label}%
```

Set the sort key for this entry:

```
2100 \@gls@defsort{\@glo@type}{\@glo@label}%

2101 \def@glo@@symbol{\@glo@text}%
2102 \ifx@glo@symbol@glo@@symbol
2103 \let@glo@symbol@glo@text
2104 \fi
2105 \gls@assign@field{relax}{\@glo@label}{symbol}{\@glo@symbol}%
2106 \expandafter
2107 \gls@assign@field\expandafter
2108 {\csname glo@\@glo@label @symbol\endcsname}
2109 {\@glo@label}{symbolplural}{\@glo@symbolplural}%
```

Define an associated boolean variable to determine whether this entry has been used yet (needs to be defined globally):

```
2110 \expandafter\xdef\csname glo@\@glo@label @flagfalse\endcsname{%
2111 \noexpand\global
2112 \noexpand\let\expandafter\noexpand
2113 \csname ifglo@\@glo@label @flag\endcsname\noexpand\iffalse
2114 }%
2115 \expandafter\xdef\csname glo@\@glo@label @flagtrue\endcsname{%
2116 \noexpand\global
2117 \noexpand\let\expandafter\noexpand
2118 \csname ifglo@\@glo@label @flag\endcsname\noexpand\iftrue
2119 }%
2120 \csname glo@\@glo@label @flagfalse\endcsname
```

Sort out any cross-referencing if required.

```
2121 \ifdefvoid\@glo@see
2122 {}%
2123 {%
2124 \protected@edef\@do@glsee{%
2125 \noexpand\@gls@fixbraces\noexpand\@glo@list\@glo@see
2126 \noexpand\@nil
2127 \noexpand\expandafter\noexpand\@glsee\noexpand\@glo@list{\@glo@label}}%
2128 \@do@glsee
2129 }%
```

Determine and store main part of the entry's index format.

```
2130 \ifignoredglossary\@glo@type
2131 {%
2132 \csdef{glo@\@glo@label @index}{}%
2133 }
2134 {%
2135 \do@glo@storeentry{\@glo@label}%
2136 }%
```

Add end hook in case another package wants to add extra keys.

```
2137 \@newglossaryentryposthook
2138 }
```

`\glossaryentryprehook` Allow extra information to be added to glossary entries:

```
2139 \newcommand*\@newglossaryentryprehook{}
```

`\glossaryentryposthook` Allow extra information to be added to glossary entries:

```
2140 \newcommand*\@newglossaryentryposthook{}
```

`\glsmoveentry` Moves entry whose label is given by first argument to the glossary named in the second argument.

```
2141 \newcommand*\glsmoveentry}[2]{%
2142 \edef\@glo@thislabel{\glsdetoklabel{#1}}%
2143 \edef\glo@type{\csname glo@\@glo@thislabel @type\endcsname}%
2144 \def\glo@list{,%}%
2145 \for@gl@entries[\@glo@type]{\@glo@label}%
2146 {%
2147 \ifdefequal\@glo@thislabel\glo@label
2148 {}{\eappto\glo@list{\glo@label,}}%
2149 }%
2150 \cslet{glo@list@\@glo@type}{\glo@list}%
2151 \csdef{glo@\@glo@thislabel @type}{#2}%
2152 }
```

`\@glossaryentryfield` Indicate what command should be used to display each entry in the glossary. (This enables the `glossaries-accsupp` package to use `\accsuppglossaryentryfield` instead.)

```

2153 \ifglxindy
2154   \newcommand*{\@glossaryentryfield}{\string\glossentry}
2155 \else
2156   \newcommand*{\@glossaryentryfield}{\string\glossentry}
2157 \fi

```

`\glossarysubentryfield` Indicate what command should be used to display each subentry in the glossary. (This enables the `glossaries-accsupp` package to use `\accsuppglossarysubentryfield` instead.)

```

2158 \ifglxindy
2159   \newcommand*{\@glossarysubentryfield}{%
2160     \string\subglossentry}
2161 \else
2162   \newcommand*{\@glossarysubentryfield}{%
2163     \string\subglossentry}
2164 \fi

```

`\@glo@storeentry` `\@glo@storeentry{<label>}`

Determine the format to write the entry in the glossary output (`.glo`) file. The argument is the entry's label (should already have been de-tok'ed if required). The result is stored in `\glo@<label>@index`, where `<label>` is the entry's label. (This doesn't include any formatting or location information.)

```
2165 \newcommand{\@glo@storeentry}[1]{%
```

Escape `makeindex/xindy` special characters in the label:

```

2166   \edef\@glo@esclabel{#1}%
2167   \@gls@checkmkidxchars\@glo@esclabel

```

Get the sort string and escape any special characters

```

2168   \protected@edef\@glo@sort{\csname glo@#1@sort\endcsname}%
2169   \@gls@checkmkidxchars\@glo@sort

```

Same again for the name string. Escape any special characters in the prefix

```
2170   \@gls@checkmkidxchars\@glo@prefix
```

Get the parent, if one exists

```
2171   \edef\@glo@parent{\csname glo@#1@parent\endcsname}%
```

Write the information to the glossary file.

```
2172   \ifglxindy
```

Store using `xindy` syntax.

```
2173     \ifx\@glo@parent\@empty
```

Entry doesn't have a parent

```

2174       \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2175         (\string"\@glo@sort\string" %
2176         \string"\@glo@prefix\@glossaryentryfield{\@glo@esclabel}\string") %
2177       }%
2178   \else

```

```

Entry has a parent
2179     \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2180         \csname glo@\@glo@parent @index\endcsname
2181         (\string"\@glo@sort\string" %
2182         \string"\@glo@prefix\@glossarysubentryfield
2183         {\csname glo@#1@level\endcsname}{\@glo@esclabel}\string") %
2184     }%
2185     \fi
2186     \else

Store using makeindex syntax.
2187     \ifx\@glo@parent\@empty

Sanitize \@glo@prefix
2188     \@onelevel@sanitize\@glo@prefix

Entry doesn't have a parent
2189     \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2190         \@glo@sort\@gls@actualchar\@glo@prefix
2191         \@glossaryentryfield{\@glo@esclabel}%
2192     }%
2193     \else

Entry has a parent
2194     \expandafter\protected@xdef\csname glo@#1@index\endcsname{%
2195         \csname glo@\@glo@parent @index\endcsname\@gls@levelchar
2196         \@glo@sort\@gls@actualchar\@glo@prefix
2197         \@glossarysubentryfield
2198         {\csname glo@#1@level\endcsname}{\@glo@esclabel}%
2199     }%
2200     \fi
2201     \fi
2202 }

```

1.8 Resetting and unsetting entry flags

Each glossary entry is assigned a conditional of the form `\ifglo@<label>@flag` which determines whether or not the entry has been used (see also `\ifglsused` defined below). These flags can be set and unset using the following macros, but first we need to know if we're in `amsmath`'s `align` environment's measuring pass.

```
\gls@ifnotmeasuring
```

```

2203 \AtBeginDocument{%
2204     \@ifpackageloaded{amsmath}%
2205     {\let\gls@ifnotmeasuring\@gls@ifnotmeasuring}%
2206     }%
2207 }
2208 \newcommand*\@gls@ifnotmeasuring[1]{%
2209     \ifmeasuring@

```

```

2210 \else
2211   #1%
2212 \fi
2213 }
2214 \newcommand*\gls@ifnotmeasuring[1]{#1}

```

`\glsreset` The command `\glsreset{<label>}` can be used to set the entry flag to indicate that it hasn't been used yet. The required argument is the entry label.

```

2215 \newcommand*\glsreset}[1]{%
2216   \gls@ifnotmeasuring
2217   {%
2218     \glsdoifexists{#1}%
2219     {%
2220       \expandafter\global\csname glo@\glsdetoklabel{#1}@flagfalse\endcsname
2221     }%
2222   }%
2223 }

```

`\glslocalreset` As above, but with only a local effect:

```

2224 \newcommand*\glslocalreset}[1]{%
2225   \gls@ifnotmeasuring
2226   {%
2227     \glsdoifexists{#1}%
2228     {%
2229       \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\iffalse
2230     }%
2231   }%
2232 }

```

`\glsunset` The command `\glsunset{<label>}` can be used to set the entry flag to indicate that it has been used. The required argument is the entry label.

```

2233 \newcommand*\glsunset}[1]{%
2234   \gls@ifnotmeasuring
2235   {%
2236     \glsdoifexists{#1}%
2237     {%
2238       \expandafter\global\csname glo@\glsdetoklabel{#1}@flagtrue\endcsname
2239     }%
2240   }%
2241 }

```

`\glslocalunset` As above, but with only a local effect:

```

2242 \newcommand*\glslocalunset}[1]{%
2243   \gls@ifnotmeasuring
2244   {%
2245     \glsdoifexists{#1}%
2246     {%
2247       \expandafter\let\csname ifglo@\glsdetoklabel{#1}@flag\endcsname\ifftrue

```

```

2248 }%
2249 }%
2250 }

```

Reset all entries for the named glossaries (supplied in a comma-separated list).
 Syntax: `\glsresetall` [*glossary-list*]

`\glsresetall`

```

2251 \newcommand*\glsresetall}[1][\@glo@types]{%
2252   \forallglsentries[#1]{\@glsentry}%
2253   {%
2254     \glsreset{\@glsentry}%
2255   }%
2256 }

```

As above, but with only a local effect:

`\glslocalresetall`

```

2257 \newcommand*\glslocalresetall}[1][\@glo@types]{%
2258   \forallglsentries[#1]{\@glsentry}%
2259   {%
2260     \glslocalreset{\@glsentry}%
2261   }%
2262 }

```

Unset all entries for the named glossaries (supplied in a comma-separated list).
 Syntax: `\glsunsetall` [*glossary-list*]

`\glsunsetall`

```

2263 \newcommand*\glsunsetall}[1][\@glo@types]{%
2264   \forallglsentries[#1]{\@glsentry}%
2265   {%
2266     \glsunset{\@glsentry}%
2267   }%
2268 }

```

As above, but with only a local effect:

`\glslocalunsetall`

```

2269 \newcommand*\glslocalunsetall}[1][\@glo@types]{%
2270   \forallglsentries[#1]{\@glsentry}%
2271   {%
2272     \glslocalunset{\@glsentry}%
2273   }%
2274 }

```

1.9 Loading files containing glossary entries

Glossary entries can be defined in an external file. These external files can contain `\newglossaryentry` and `\newacronym` commands.¹

¹and any other valid \LaTeX code that can be used in the preamble.

`\loadglsentries[⟨type⟩]{⟨filename⟩}`

This command will input the file using `\input`. The optional argument specifies to which glossary the entries should be assigned if they haven't used the type key. If the optional argument is not specified, the default glossary is used. Only those entries used in the document (via `\glslink`, `\gls`, `\glspl` and uppercase variants or `\glsadd` and `\glsaddall` will appear in the glossary). The mandatory argument is the filename (with or without `.tex` extension).

`\loadglsentries`

```
2275 \newcommand*\loadglsentries}[2][\@gls@default]{%
2276   \let\@gls@default\glsdefaulttype
2277   \def\glsdefaulttype{#1}\input{#2}%
2278   \let\glsdefaulttype\@gls@default
2279 }
```

`\loadglsentries` can only be used in the preamble:

```
2280 \@onlypreamble{\loadglsentries}
```

1.10 Using glossary entries in the text

Any term that has been defined using `\newglossaryentry` (or `\newacronym`) can be displayed in the text (i.e. outside of the glossary) using one of the commands defined in this section. Unless you use `\glslink`, the way the term appears in the text is determined by `\glsdisplayfirst` (if it is the first time the term has been used) or `\glsdisplay` (for subsequent use). Any formatting commands (such as `\textbf`) is governed by `\glstextformat`. By default this just displays the link text “as is”.

`\glstextformat`

```
2281 \newcommand*\glstextformat}[1]{#1}
```

`\glsentryfmt` As from version 3.11a, the way in which an entry is displayed is now governed by `\glsentryfmt`. This doesn't take any arguments. The required information is set by commands like `\gls`. To ensure backward compatibility, the default use the old `\glsdisplay` and `\glsdisplayfirst` style of commands

```
2282 \newcommand*\glsentryfmt{%
2283   \@@gls@default@entryfmt\glsdisplayfirst\glsdisplay
2284 }
```

Format that provides backwards compatibility:

```
2285 \newcommand*\@@gls@default@entryfmt}[2]{%
2286   \ifdefempty\glscustomtext
2287   {%
2288     \glsifplural
2289     {%
```

Plural form

```
2290     \glscapscase
2291     {%
```

Don't adjust case

```
2292     \ifglused\glslabel
2293     {%
```

Subsequent use

```
2294     #2{\glsentryplural{\glslabel}}%
2295     {\glsentrydescplural{\glslabel}}%
2296     {\glsentrysymbolplural{\glslabel}}{\glsinsert}%
2297     }%
2298     {%
```

First use

```
2299     #1{\glsentryfirstplural{\glslabel}}%
2300     {\glsentrydescplural{\glslabel}}%
2301     {\glsentrysymbolplural{\glslabel}}{\glsinsert}%
2302     }%
2303     }%
2304     {%
```

Make first letter upper case

```
2305     \ifglused\glslabel
2306     {%
```

Subsequent use. (Expansion was used in version 3.07 and below in case the name wasn't the first thing to be displayed, but now the user can sort out the upper casing in `\defglsentryfmt`, which avoids the issues caused by fragile commands.)

```
2307     \ifbool{glscompatible-3.07}%
2308     {%
2309     \protected@edef\@glo@etext{%
2310     #2{\glsentryplural{\glslabel}}%
2311     {\glsentrydescplural{\glslabel}}%
2312     {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2313     \xmakefirstuc\@glo@etext
2314     }%
2315     {%
2316     #2{\Glsentryplural{\glslabel}}%
2317     {\glsentrydescplural{\glslabel}}%
2318     {\glsentrysymbolplural{\glslabel}}{\glsinsert}%
2319     }%
2320     }%
2321     {%
```

First use

```
2322     \ifbool{glscompatible-3.07}%
2323     {%
2324     \protected@edef\@glo@etext{%
```

```

2325         #1{\glsentryfirstplural{\glslabel}}%
2326         {\glsentrydescplural{\glslabel}}%
2327         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2328     \xmakefirstuc\@glo@etext
2329     }%
2330     {%
2331         #1{\Glsentryfirstplural{\glslabel}}%
2332         {\glsentrydescplural{\glslabel}}%
2333         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2334     }%
2335     }%
2336     }%
2337     {%

```

Make all upper case

```

2338     \ifglsused\glslabel
2339     {%

```

Subsequent use

```

2340         \mfirstucMakeUppercase{#2{\glsentryplural{\glslabel}}%
2341         {\glsentrydescplural{\glslabel}}%
2342         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2343     }%
2344     {%

```

First use

```

2345         \mfirstucMakeUppercase{#1{\glsentryfirstplural{\glslabel}}%
2346         {\glsentrydescplural{\glslabel}}%
2347         {\glsentrysymbolplural{\glslabel}}{\glsinsert}}%
2348     }%
2349     }%
2350     }%
2351     {%

```

Singular form

```

2352     \gls caps case
2353     {%

```

Don't adjust case

```

2354     \ifglsused\glslabel
2355     {%

```

Subsequent use

```

2356         #2{\glsentrytext{\glslabel}}%
2357         {\glsentrydesc{\glslabel}}%
2358         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2359     }%
2360     {%

```

First use

```

2361         #1{\glsentryfirst{\glslabel}}%
2362         {\glsentrydesc{\glslabel}}%

```

```

2363         {\glsentrysymbol{\glslabel}}{\glsinsert}%
2364     }%
2365 }%
2366 {%

```

Make first letter upper case

```

2367     \ifglsused\glslabel
2368     {%

```

Subsequent use

```

2369     \ifbool{glscompatible-3.07}%
2370     {%
2371         \protected@edef\@glo@etext{%
2372             #2{\glsentrytext{\glslabel}}%
2373             {\glsentrydesc{\glslabel}}%
2374             {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2375         \xmakefirstuc\@glo@etext
2376     }%
2377     {%
2378         #2{\Glsentrytext{\glslabel}}%
2379         {\glsentrydesc{\glslabel}}%
2380         {\glsentrysymbol{\glslabel}}{\glsinsert}%
2381     }%
2382 }%
2383 {%

```

First use

```

2384     \ifbool{glscompatible-3.07}%
2385     {%
2386         \protected@edef\@glo@etext{%
2387             #1{\glsentryfirst{\glslabel}}%
2388             {\glsentrydesc{\glslabel}}%
2389             {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2390         \xmakefirstuc\@glo@etext
2391     }%
2392     {%
2393         #1{\Glsentryfirst{\glslabel}}%
2394         {\glsentrydesc{\glslabel}}%
2395         {\glsentrysymbol{\glslabel}}{\glsinsert}%
2396     }%
2397 }%
2398 }%
2399 {%

```

Make all upper case

```

2400     \ifglsused\glslabel
2401     {%

```

Subsequent use

```

2402         \mfirstucMakeUppercase{#2{\glsentrytext{\glslabel}}%
2403         {\glsentrydesc{\glslabel}}%

```

```

2404         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2405     }%
2406     {%

```

First use

```

2407         \mfirstucMakeUppercase{#1{\glsentryfirst{\glslabel}}}%
2408         {\glsentrydesc{\glslabel}}}%
2409         {\glsentrysymbol{\glslabel}}{\glsinsert}}%
2410     }%
2411 }%
2412 }%
2413 }%
2414 {%

```

Custom text provided in \glsdisp

```

2415     \ifglsused{\glslabel}%
2416     {%

```

Subsequent use

```

2417         #2{\glscustomtext}%
2418         {\glsentrydesc{\glslabel}}}%
2419         {\glsentrysymbol{\glslabel}}}{}%
2420     }%
2421     {%

```

First use

```

2422         #1{\glscustomtext}%
2423         {\glsentrydesc{\glslabel}}}%
2424         {\glsentrysymbol{\glslabel}}}{}%
2425     }%
2426 }%
2427 }

```

`\glsgenentryfmt` Define a generic format that just uses the first, text, plural or first plural keys (or the custom text) with the insert text appended.

```

2428 \newcommand*{\glsgenentryfmt}{%
2429     \ifdefempty\glscustomtext
2430     {%
2431         \glsifplural
2432         {%

```

Plural form

```

2433         \glsupcase
2434         {%

```

Don't adjust case

```

2435         \ifglsused\glslabel
2436         {%

```

Subsequent use

```

2437         \glsentryplural{\glslabel}\glsinsert
2438     }%
2439     {%

```

First use

```
2440      \glstentryfirstplural{\glslabel}\glsinsert
2441      }%
2442      }%
2443      {%
```

Make first letter upper case

```
2444      \ifglsused\glslabel
2445      {%
```

Subsequent use.

```
2446      \Glstentryplural{\glslabel}\glsinsert
2447      }%
2448      {%
```

First use

```
2449      \Glstentryfirstplural{\glslabel}\glsinsert
2450      }%
2451      }%
2452      {%
```

Make all upper case

```
2453      \ifglsused\glslabel
2454      {%
```

Subsequent use

```
2455      \mfirstucMakeUppercase
2456      {\glstentryplural{\glslabel}\glsinsert}%
2457      }%
2458      {%
```

First use

```
2459      \mfirstucMakeUppercase
2460      {\glstentryfirstplural{\glslabel}\glsinsert}%
2461      }%
2462      }%
2463      }%
2464      {%
```

Singular form

```
2465      \glscapscase
2466      {%
```

Don't adjust case

```
2467      \ifglsused\glslabel
2468      {%
```

Subsequent use

```
2469      \glstentrytext{\glslabel}\glsinsert
2470      }%
2471      {%
```

First use

```
2472     \glsentryfirst{\glslabel}\glsinsert
2473     }%
2474     }%
2475     {%
```

Make first letter upper case

```
2476     \ifglsused\glslabel
2477     {%
```

Subsequent use

```
2478     \Glsentrytext{\glslabel}\glsinsert
2479     }%
2480     {%
```

First use

```
2481     \Glsentryfirst{\glslabel}\glsinsert
2482     }%
2483     }%
2484     {%
```

Make all upper case

```
2485     \ifglsused\glslabel
2486     {%
```

Subsequent use

```
2487     \mfirstucMakeUppercase{\glsentrytext{\glslabel}\glsinsert}%
2488     }%
2489     {%
```

First use

```
2490     \mfirstucMakeUppercase{\glsentryfirst{\glslabel}\glsinsert}%
2491     }%
2492     }%
2493     }%
2494     }%
2495     {%
```

Custom text provided in `\glsdisp`. (The insert is most likely to be empty at this point.)

```
2496     \glscustomtext\glsinsert
2497     }%
2498 }
```

`\glsgenacfmt` Define a generic acronym format that uses the long and short keys (or their plurals) and `\acrfullformat`, `\firstacronymfont` and `\acronymfont`.

```
2499 \newcommand*{\glsgenacfmt}{%
2500   \ifdefempty\glscustomtext
2501   {%
2502     \ifglsused\glslabel
2503     {%
```

Subsequent use:

2504 \glsifplural
2505 {%

Subsequent plural form:

2506 \glsifplural
2507 {%

Subsequent plural form, don't adjust case:

2508 \acronymfont{\glsentryshortpl{\glslabel}}\glsinsert
2509 }%
2510 {%

Subsequent plural form, make first letter upper case:

2511 \acronymfont{\Glsentryshortpl{\glslabel}}\glsinsert
2512 }%
2513 {%

Subsequent plural form, all caps:

2514 \mfirstucMakeUppercase
2515 {\acronymfont{\glsentryshortpl{\glslabel}}\glsinsert}%
2516 }%
2517 }%
2518 {%

Subsequent singular form

2519 \glsifplural
2520 {%

Subsequent singular form, don't adjust case:

2521 \acronymfont{\glsentryshort{\glslabel}}\glsinsert
2522 }%
2523 {%

Subsequent singular form, make first letter upper case:

2524 \acronymfont{\Glsentryshort{\glslabel}}\glsinsert
2525 }%
2526 {%

Subsequent singular form, all caps:

2527 \mfirstucMakeUppercase
2528 {\acronymfont{\glsentryshort{\glslabel}}\glsinsert}%
2529 }%
2530 }%
2531 }%
2532 {%

First use:

2533 \glsifplural
2534 {%

First use plural form:

2535 \glsifplural
2536 {%

First use plural form, don't adjust case:

```
2537      \genplacrfullformat{\glslabel}{\glsinsert}%  
2538      }%  
2539      {%
```

First use plural form, make first letter upper case:

```
2540      \Genplacrfullformat{\glslabel}{\glsinsert}%  
2541      }%  
2542      {%
```

First use plural form, all caps:

```
2543      \mfirstucMakeUppercase  
2544      {\genplacrfullformat{\glslabel}{\glsinsert}}%  
2545      }%  
2546      }%  
2547      {%
```

First use singular form

```
2548      \glsapscase  
2549      {%
```

First use singular form, don't adjust case:

```
2550      \genacrfullformat{\glslabel}{\glsinsert}%  
2551      }%  
2552      {%
```

First use singular form, make first letter upper case:

```
2553      \Genacrfullformat{\glslabel}{\glsinsert}%  
2554      }%  
2555      {%
```

First use singular form, all caps:

```
2556      \mfirstucMakeUppercase  
2557      {\genacrfullformat{\glslabel}{\glsinsert}}%  
2558      }%  
2559      }%  
2560      }%  
2561      }%  
2562      {%
```

User supplied text.

```
2563      \glscustomtext  
2564      }%  
2565      }
```

`\genacrfullformat`

`\genacrfullformat{<label>}{<insert>}`

The full format used by `\glsngenacfmt` (singular).

```
2566 \newcommand*{\genacrfullformat}[2]{%  
2567   \glsentrylong{#1}#2\space
```

```
2568 (\protect\firstacronymfont{\glsentryshort{#1}})%
2569 }
```

`\Genacrfullformat` `\Genacrfullformat{<label>}{<insert>}`

As above but makes the first letter upper case.

```
2570 \newcommand*{\Genacrfullformat}[2]{%
2571   \protected@edef\gls@text{\genacrfullformat{#1}{#2}}%
2572   \xmakefirstuc\gls@text
2573 }
```

`\genplacrfullformat` `\genplacrfullformat{<label>}{<insert>}`

The full format used by `\glsngenacfmt` (plural).

```
2574 \newcommand*{\genplacrfullformat}[2]{%
2575   \glsentrylongpl{#1}#2\space
2576   (\protect\firstacronymfont{\glsentryshortpl{#1}})%
2577 }
```

`\Genplacrfullformat` `\Genplacrfullformat{<label>}{<insert>}`

As above but makes the first letter upper case.

```
2578 \newcommand*{\Genplacrfullformat}[2]{%
2579   \protected@edef\gls@text{\genplacrfullformat{#1}{#2}}%
2580   \xmakefirstuc\gls@text
2581 }
```

`\glsdisplayfirst` Deprecated. Kept for backward compatibility.
2582 \newcommand*{\glsdisplayfirst}[4]{#1#4}

`\glsdisplay` Deprecated. Kept for backward compatibility.
2583 \newcommand*{\glsdisplay}[4]{#1#4}

`\defglsdisplay` Deprecated. Kept for backward compatibility.

```
2584 \newcommand*{\defglsdisplay}[2][\glsdefaulttype]{%
2585   \GlossariesWarning{\string\defglsdisplay\space is now obsolete.^^J
2586   Use \string\defglsentryfmt\space instead}%
2587   \expandafter\def\csname gls@#1@display\endcsname##1##2##3##4{#2}%
2588   \edef@gls@doentrydef{%
2589     \noexpand\defglsentryfmt[#1]{%
2590       \noexpand\ifcsdef{gls@#1@displayfirst}%
2591       {%
2592         \noexpand@@gls@default@entryfmt
2593         {\noexpand\csuse{gls@#1@displayfirst}}%

```

```

2594         {\noexpand\csuse{gls@#1@display}}%
2595     }%
2596     {%
2597         \noexpand\@gls@default@entryfmt
2598         {\noexpand\glsdisplayfirst}%
2599         {\noexpand\csuse{gls@#1@display}}%
2600     }%
2601 }%
2602 }%
2603 \@gls@doentrydef
2604 }

```

`\defglsdisplayfirst` Deprecated. Kept for backward compatibility.

```

2605 \newcommand*{\defglsdisplayfirst}[2][\glsdefaultttype]{%
2606   \GlossariesWarning{\string\defglsdisplayfirst\space is now obsolete.^^J
2607   Use \string\defglsentryfmt\space instead}%
2608   \expandafter\def\csname gls@#1@displayfirst\endcsname##1##2##3##4{#2}%
2609   \edef\@gls@doentrydef{%
2610     \noexpand\defglsentryfmt[#1]{%
2611       \noexpand\ifcsdef{gls@#1@display}%
2612       {%
2613         \noexpand\@gls@default@entryfmt
2614         {\noexpand\csuse{gls@#1@displayfirst}}%
2615         {\noexpand\csuse{gls@#1@display}}%
2616       }%
2617       {%
2618         \noexpand\@gls@default@entryfmt
2619         {\noexpand\csuse{gls@#1@displayfirst}}%
2620         {\noexpand\glsdisplay}%
2621       }%
2622     }%
2623   }%
2624   \@gls@doentrydef
2625 }

```

1.10.1 Links to glossary entries

The links to glossary entries all have a first optional argument that can be used to change the format and counter of the associated entry number. Except for `\glslink` and `\glsdisp`, the commands like `\gls` have a final optional argument that can be used to insert additional text in the link (this will usually be appended, but can be redefined using `\defentryfmt`). It goes against the \LaTeX norm to have an optional argument after the mandatory arguments, but it makes more sense to write, say, `\gls{label}[’s]` rather than, say, `\gls[append=’s]{label}`. Since these control sequences are defined to include the final square bracket, spaces will be ignored after them. This is likely

to lead to confusion as most users would not expect, say, `\gls{<label>}` to ignore following spaces, so `\new@ifnextchar` from the package is required.

The following keys can be used in the first optional argument. The counter key checks that the value is the name of a valid counter.

```

2626 \define@key{glslink}{counter}{%
2627   \ifcsundef{c@#1}%
2628   {%
2629     \PackageError{glossaries}%
2630     {There is no counter called ‘#1’}%
2631     {%
2632       The counter key should have the name of a valid counter
2633       as its value%
2634     }%
2635   }%
2636   {%
2637     \def\@gls@counter{#1}%
2638   }%
2639 }

```

The value of the format key should be the name of a command (without the initial backslash) that has a single mandatory argument which can be used to format the associated entry number.

```

2640 \define@key{glslink}{format}{%
2641   \def\@glsnumberformat{#1}}

```

The hyper key is a boolean key, it can either have the value true or false, and indicates whether or not to make a hyperlink to the relevant glossary entry. If hyper is false, an entry will still be made in the glossary, but the given text won't be a hyperlink.

```

2642 \define@boolkey{glslink}{hyper}[true]{}

```

Initialise hyper key.

```

2643 \ifdef{\hyperlink}{\KV@glslink@hypertrue}{\KV@glslink@hyperfalse}

```

The local key is a boolean key. If true this indicates that commands such as `\gls` should only do a local reset rather than a global one.

```

2644 \define@boolkey{glslink}{local}[true]{}

```

The original `\glsifhyper` command isn't particularly useful as it makes more sense to check the actual hyperlink setting rather than testing whether the starred or unstarred version has been used. Therefore, as from version 4.08, `\glsifhyper` is deprecated in favour of `\glsifhyperon`. In case there is a particular need to know whether the starred or unstarred version was used, provide a new command that determines whether the *-version, +-version or unmodified version was used.

<code>\glslinkvar{<unmodified case>}{<star case>}{<plus case>}</code>

`\glslinkvar` Initialise to unmodified case.

```
2645 \newcommand*{\glslinkvar}[3]{#1}
```

`\glsifhyper` Now deprecated.

```
2646 \newcommand*{\glsifhyper}[2]{%
2647 \glslinkvar{#1}{#2}{#1}%
2648 \GlossariesWarning{\string\glsifhyper\space is deprecated. Did
2649 you mean \string\glsifhyperon\space or \string\glslinkvar?}%
2650 }
```

`\@gls@hyp@opt` Used by the commands such as `\glslink` to determine whether to modify the hyper option.

```
2651 \newcommand*{\@gls@hyp@opt}[1]{%
2652 \let\glslinkvar\@firstofthree
2653 \let\@gls@hyp@opt@cs#1\relax
2654 \ifstar{\s@gls@hyp@opt}%
2655 {\@ifnextchar+{\@firstoftwo{\p@gls@hyp@opt}}{#1}}%
2656 }
```

`\s@gls@hyp@opt` Starred version

```
2657 \newcommand*{\s@gls@hyp@opt}[1] []{%
2658 \let\glslinkvar\@secondofthree
2659 \@gls@hyp@opt@cs[hyper=false,#1]}
```

`\p@gls@hyp@opt` Plus version

```
2660 \newcommand*{\p@gls@hyp@opt}[1] []{%
2661 \let\glslinkvar\@thirdofthree
2662 \@gls@hyp@opt@cs[hyper=true,#1]}
```

Syntax:

```
\glslink[<options>]{<label>}{<text>}
```

Display *<text>* in the document, and add the entry information for *<label>* into the relevant glossary. The optional argument should be a key value list using the `\glslink` keys defined above.

There is also a starred version:

```
\glslink* [<options>]{<label>}{<text>}
```

which is equivalent to `\glslink[hyper=false, <options>]{<label>}{<text>}`

First determine which version is being used:

`\glslink`

```
2663 \newrobustcmd*{\glslink}{%
2664 \@gls@hyp@opt\@gls@@link
2665 }
```

`\@gls@link` The main part of the business is in `\@gls@link` which shouldn't check if the term is defined as it's called by `\gls` etc which also perform that check.

```
2666 \newcommand*\@gls@link}[3] []{%
2667   \ifglsentryexists{#2}%
2668   {%
2669     \let\do@gls@link@checkfirsthyper\relax
2670     \@gls@link[#1]{#2}{#3}%
2671   }{%
2672     \PackageError{glossaries}{Glossary entry ‘#2’ has not been
2673     defined}{You need to define a glossary entry before you
2674     can use it.}%
```

Display the specified text. (The entry doesn't exist so there's nothing to link it to.)

```
2675   \glstextformat{#3}%
2676 }%
2677 }
```

`\@gls@link@checkfirsthyper` Check for first use and switch off hyper key if hyperlink not wanted. (Should be off if first use and `hyper=false` is on or if first use and both the entry is in an acronym list and the `acrfootnote` setting is on.) This assumes the glossary type is stored in `\glsstype` and the label is stored in `\glslabel`.

```
2678 \newcommand*\@gls@link@checkfirsthyper{%
2679   \ifglsused{\glslabel}%
2680   {%
2681   }%
2682   {%
2683     \gls@checkisacronymlist\glsstype
2684     \ifglshyperfirst
2685       \ifglsisacronymlist
2686         \ifglsacrfootnote
2687           \KV@glslink@hyperfalse
2688           \fi
2689         \fi
2690       \else
2691         \KV@glslink@hyperfalse
2692         \fi
2693   }%
```

Allow user to hook into this

```
2694   \glslinkcheckfirsthyperhook
2695 }
```

`\@gls@link@checkfirsthyperhook` Allow used to hook into the `\@gls@link@checkfirsthyper` macro

```
2696 \newcommand*\@glslinkcheckfirsthyperhook{}
```

`\@gls@link`

```
2697 \def\@gls@link[#1]#2#3{%
```

Inserting `\leavevmode` suggested by Donald Arseneau (avoids problem with `tabularx`).

```
2698 \leavevmode
2699 \edef\glslabel{\glsdetoklabel{#2}}%

Save options in \@gls@link@opts and label in \@gls@link@label
2700 \def\@gls@link@opts{#1}%
2701 \let\@gls@link@label\glslabel

2702 \def\@glsnumberformat{glsnumberformat}%
2703 \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
```

If this is in one of the “nohypertypes” glossaries, suppress the hyperlink by default

```
2704 \edef\glsstype{\csname glo@\glslabel @type\endcsname}%
```

Save original setting

```
2705 \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper
```

Switch off hyper setting if the glossary type has been identified in `nohyperlist`.

```
2706 \expandafter\DTLifinlist\expandafter
2707   {\glsstype}{\@gls@nohyperlist}%
2708   {%
2709     \KV@glslink@hyperfalse
2710   }%
2711   {%
2712   }%
```

Macros must set this before calling `\@gls@link`. The commands that check the first use flag should set this to `\@gls@link@checkfirsthyper` otherwise it should be set to `\relax`.

```
2713 \do@gls@link@checkfirsthyper
2714 \setkeys{glslink}{#1}%
```

Define `\glsifhyperon`

```
2715 \ifKV@glslink@hyper
2716   \let\glsifhyperon\@firstoftwo
2717 \else
2718   \let\glsifhyperon\@secondoftwo
2719 \fi
```

Store the entry’s counter in `\theglentrycounter`

```
2720 \@gls@saveentrycounter
```

Define sort key if necessary:

```
2721 \@gls@setsort{\glslabel}%
```

(De-tok’ing done by `\@do@wrglossary`)

```
2722 \@do@wrglossary{#2}%
2723 \ifKV@glslink@hyper
2724   \@glslink{\glo@linkprefix\glslabel}{\glstextformat{#3}}%
2725 \else
```

```
2726     \glstextformat{#3}%
2727     \fi
```

Restore original setting

```
2728     \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
2729 }
```

`\glolinkprefix`

```
2730 \newcommand*{\glolinkprefix}{glo:}
```

`\glentrycounter` Set default value of entry counter

```
2731 \def\glentrycounter{\glscounter}%
```

`\glsaveentrycounter` Need to check if using equation counter in align environment:

```
2732 \newcommand*{\@gls@saveentrycounter}{%
```

```
2733   \def\@gls@Hcounter{}}%
```

Are we using equation counter?

```
2734   \ifthenelse{\equal{\@gls@counter}{equation}}{%
```

```
2735     {
```

If we're in align environment, `\xatlevel@` will be defined. (Can't test for `\@currentenv` as may be inside an inner environment.)

```
2736     \ifcsundef{xatlevel@}%
```

```
2737     {%
```

```
2738       \edef\theglentrycounter{\expandafter\noexpand
```

```
2739         \csname the\@gls@counter\endcsname}%
```

```
2740     }%
```

```
2741     {%
```

```
2742       \ifx\xatlevel@\@empty
```

```
2743         \edef\theglentrycounter{\expandafter\noexpand
```

```
2744           \csname the\@gls@counter\endcsname}%
```

```
2745       \else
```

```
2746         \savecounters@
```

```
2747         \advance\c@equation by 1\relax
```

```
2748         \edef\theglentrycounter{\csname the\@gls@counter\endcsname}%
```

Check if hyperref version of this counter

```
2749     \ifcsundef{theH\@gls@counter}%
```

```
2750     {%
```

```
2751       \def\@gls@Hcounter{\theglentrycounter}%
```

```
2752     }%
```

```
2753     {%
```

```
2754       \def\@gls@Hcounter{\csname theH\@gls@counter\endcsname}%
```

```
2755     }%
```

```
2756     \protected@edef\theHglentrycounter{\@gls@Hcounter}%
```

```
2757     \restorecounters@
```

```
2758     \fi
```

```
2759   }%
```

```
2760 }%
```

```
2761 {%
```


Not using equation counter so no special measures:

```
2762 \edef\theglsentrycounter{\expandafter\noexpand
2763 \csname the\@gls@counter\endcsname}%
2764 }%
```

Check if hyperref version of this counter

```
2765 \ifx\@gls@Hcounter\@empty
2766 \ifcsundef{theH\@gls@counter}%
2767 {%
2768 \def\theHglsentrycounter{\theglsentrycounter}%
2769 }%
2770 {%
2771 \protected@edef\theHglsentrycounter{\expandafter\noexpand
2772 \csname theH\@gls@counter\endcsname}%
2773 }%
2774 \fi
2775 }
```

`\@set@glo@numformat` Set the formatting information in the format required by `makeindex`. The first argument is the format specified by the user (via the `format` key), the second argument is the name of the counter used to indicate the location, the third argument is a control sequence which stores the required format and the fourth argument (new to v3.0) is the hyper-prefix.

```
2776 \def\@set@glo@numformat#1#2#3#4{%
2777 \expandafter\@glo@check@mkidrangechar#3\@nil
2778 \protected@edef#1{%
2779 \@glo@prefix setentrycounter[#4]{#2}%
2780 \expandafter\string\csname\@glo@suffix\endcsname
2781 }%
2782 \@gls@checkmkidchars#1%
2783 }
```

Check to see if the given string starts with a (or). If it does set `\@glo@prefix` to the starting character, and `\@glo@suffix` to the rest (or `glsnumberformat` if there is nothing else), otherwise set `\@glo@prefix` to nothing and `\@glo@suffix` to all of it.

```
2784 \def\@glo@check@mkidrangechar#1#2\@nil{%
2785 \if#1(\relax
2786 \def\@glo@prefix{(%
2787 \if\relax#2\relax
2788 \def\@glo@suffix{glsnumberformat}%
2789 \else
2790 \def\@glo@suffix{#2}%
2791 \fi
2792 \else
2793 \if#1)\relax
2794 \def\@glo@prefix{)}%
2795 \if\relax#2\relax
```

```

2796     \def\@glo@suffix{glsnumberformat}%
2797     \else
2798     \def\@glo@suffix{#2}%
2799     \fi
2800     \else
2801     \def\@glo@prefix{ }\def\@glo@suffix{#1#2}%
2802     \fi
2803 \fi}

```

`\@gls@escbsdq` Escape backslashes and double quote marks. The argument must be a control sequence.

```

2804 \newcommand*{\@gls@escbsdq}[1]{%
2805   \def\@gls@checkedmkidx{ }%
2806   \let\gls@xdystring=#1\relax
2807   \@onelevel@sanitize\gls@xdystring
2808   \edef\do@gls@xdycheckbackslash{%
2809     \noexpand\@gls@xdycheckbackslash\gls@xdystring\noexpand\@nil
2810     \@backslashchar\@backslashchar\noexpand\null}%
2811   \do@gls@xdycheckbackslash
2812   \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%
2813   \def\@gls@checkedmkidx{ }%
2814   \expandafter\@gls@xdycheckquote\gls@xdystring\@nil""\null
2815   \expandafter\@gls@updatechecked\@gls@checkedmkidx{\gls@xdystring}%

```

Unsanitize `\gls@numberpage`, `\gls@alphpage`, `\gls@Alphpage` and `\gls@romanpage` (thanks to David Carlisle for the suggestion.)

```

2816   \@for\@gls@tmp:=\gls@protected@pagefmts\do
2817   {%
2818     \edef\@gls@sanitized@tmp{\expandafter\@gobble\string\\ \expandonce\@gls@tmp}%
2819     \@onelevel@sanitize\@gls@sanitized@tmp
2820     \edef\gls@dostubst{%
2821       \noexpand\DTLsubstituteall\noexpand\gls@xdystring
2822       {\@gls@sanitized@tmp}{\expandonce\@gls@tmp}%
2823     }%
2824     \gls@dostubst
2825   }%

```

Assign to required control sequence

```

2826   \let#1=\gls@xdystring
2827 }

```

Catch special characters (argument must be a control sequence):

`\gls@checkmkidxchars`

```

2828 \newcommand{\@gls@checkmkidxchars}[1]{%
2829   \ifglsxindy
2830     \@gls@escbsdq{#1}%
2831   \else
2832     \def\@gls@checkedmkidx{ }%
2833     \expandafter\@gls@checkquote#1\@nil""\null

```

```

2834 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2835 \def\@gls@checkedmkidx{}%
2836 \expandafter\@gls@checkescquote#1\@nil\\"\null
2837 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2838 \def\@gls@checkedmkidx{}%
2839 \expandafter\@gls@checkescactual#1\@nil\?\?\null
2840 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2841 \def\@gls@checkedmkidx{}%
2842 \expandafter\@gls@checkactual#1\@nil??\null
2843 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2844 \def\@gls@checkedmkidx{}%
2845 \expandafter\@gls@checkbar#1\@nil||\null
2846 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2847 \def\@gls@checkedmkidx{}%
2848 \expandafter\@gls@checkesbar#1\@nil\\|\null
2849 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2850 \def\@gls@checkedmkidx{}%
2851 \expandafter\@gls@checklevel#1\@nil!!\null
2852 \expandafter\@gls@updatechecked\@gls@checkedmkidx{#1}%
2853 \fi
2854 }

```

Update the control sequence and strip trailing \@nil:

\@gls@updatechecked

```
2855 \def\@gls@updatechecked#1\@nil#2{\def#2{#1}}
```

\@gls@tmpb Define temporary token

```
2856 \newtoks\@gls@tmpb
```

\@gls@checkquote Replace " with "" since " is a makeindex special character.

```

2857 \def\@gls@checkquote#1"#2"#3\null{%
2858 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2859 \toks@={#1}%
2860 \ifx\null#2\null
2861 \ifx\null#3\null
2862 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2863 \def\@gls@checkquote{\relax}%
2864 \else
2865 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2866 \@gls@quotechar\@gls@quotechar\@gls@quotechar\@gls@quotechar}%
2867 \def\@gls@checkquote{\@gls@checkquote#3\null}%
2868 \fi
2869 \else
2870 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2871 \@gls@quotechar\@gls@quotechar}%
2872 \ifx\null#3\null
2873 \def\@gls@checkquote{\@gls@checkquote#2""\null}%
2874 \else

```

```

2875 \def\@gls@checkquote{\@gls@checkquote#2"#3\null}%
2876 \fi
2877 \fi
2878 \@gls@checkquote
2879 }

```

\@gls@checkescquote Do the same for \":

```

2880 \def\@gls@checkescquote#1\"#2\"#3\null{%
2881 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2882 \toks@={#1}%
2883 \ifx\null#2\null
2884 \ifx\null#3\null
2885 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2886 \def\@gls@checkescquote{\relax}%
2887 \else
2888 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2889 \@gls@quotechar\string\"@gls@quotechar
2890 \@gls@quotechar\string\"@gls@quotechar}%
2891 \def\@gls@checkescquote{\@gls@checkescquote#3\null}%
2892 \fi
2893 \else
2894 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2895 \@gls@quotechar\string\"@gls@quotechar}%
2896 \ifx\null#3\null
2897 \def\@gls@checkescquote{\@gls@checkescquote#2\"\" \null}%
2898 \else
2899 \def\@gls@checkescquote{\@gls@checkescquote#2\"#3\null}%
2900 \fi
2901 \fi
2902 \@gls@checkescquote
2903 }

```

\@gls@checkescactual Similarly for \? (which is replaces @ as makeindex's special character):

```

2904 \def\@gls@checkescactual#1\?#2\?#3\null{%
2905 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2906 \toks@={#1}%
2907 \ifx\null#2\null
2908 \ifx\null#3\null
2909 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2910 \def\@gls@checkescactual{\relax}%
2911 \else
2912 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2913 \@gls@quotechar\string\"@gls@actualchar
2914 \@gls@quotechar\string\"@gls@actualchar}%
2915 \def\@gls@checkescactual{\@gls@checkescactual#3\null}%
2916 \fi
2917 \else
2918 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2919 \@gls@quotechar\string\"@gls@actualchar}%

```

```

2920 \ifx\null#3\null
2921 \def\@gls@checkescactual{\@gls@checkescactual#2\?\?\null}%
2922 \else
2923 \def\@gls@checkescactual{\@gls@checkescactual#2\?#3\null}%
2924 \fi
2925 \fi
2926 \@gls@checkescactual
2927 }

```

\@gls@checkescbar Similarly for \|:

```

2928 \def\@gls@checkescbar#1\|#2\|#3\null{%
2929 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2930 \toks@={#1}%
2931 \ifx\null#2\null
2932 \ifx\null#3\null
2933 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2934 \def\@gls@checkescbar{\relax}%
2935 \else
2936 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2937 \@gls@quotechar\string\\"\@gls@encapchar
2938 \@gls@quotechar\string\\"\@gls@encapchar}%
2939 \def\@gls@checkescbar{\@gls@checkescbar#3\null}%
2940 \fi
2941 \else
2942 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2943 \@gls@quotechar\string\\"\@gls@encapchar}%
2944 \ifx\null#3\null
2945 \def\@gls@checkescbar{\@gls@checkescbar#2\|\|\null}%
2946 \else
2947 \def\@gls@checkescbar{\@gls@checkescbar#2\|#3\null}%
2948 \fi
2949 \fi
2950 \@gls@checkescbar
2951 }

```

\@gls@checkesclevel Similarly for \!:

```

2952 \def\@gls@checkesclevel#1\!#2\!#3\null{%
2953 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2954 \toks@={#1}%
2955 \ifx\null#2\null
2956 \ifx\null#3\null
2957 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2958 \def\@gls@checkesclevel{\relax}%
2959 \else
2960 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2961 \@gls@quotechar\string\!\@gls@levelchar
2962 \@gls@quotechar\string\!\@gls@levelchar}%
2963 \def\@gls@checkesclevel{\@gls@checkesclevel#3\null}%
2964 \fi

```

```

2965 \else
2966 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2967 \@gls@quotechar\string"\@gls@levelchar}%
2968 \ifx\null#3\null
2969 \def\@gls@checkesclevel{\@gls@checkesclevel#2\!\!\null}%
2970 \else
2971 \def\@gls@checkesclevel{\@gls@checkesclevel#2\!#3\null}%
2972 \fi
2973 \fi
2974 \@gls@checkesclevel
2975 }

```

\@gls@checkbar and for |:

```

2976 \def\@gls@checkbar#1|#2|#3\null{%
2977 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
2978 \toks@={#1}%
2979 \ifx\null#2\null
2980 \ifx\null#3\null
2981 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
2982 \def\@gls@checkbar{\relax}%
2983 \else
2984 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2985 \@gls@quotechar\@gls@encapchar\@gls@quotechar\@gls@encapchar}%
2986 \def\@gls@checkbar{\@gls@checkbar#3\null}%
2987 \fi
2988 \else
2989 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
2990 \@gls@quotechar\@gls@encapchar}%
2991 \ifx\null#3\null
2992 \def\@gls@checkbar{\@gls@checkbar#2|\null}%
2993 \else
2994 \def\@gls@checkbar{\@gls@checkbar#2|#3\null}%
2995 \fi
2996 \fi
2997 \@gls@checkbar
2998 }

```

\@gls@checklevel and for !:

```

2999 \def\@gls@checklevel#1!#2!#3\null{%
3000 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
3001 \toks@={#1}%
3002 \ifx\null#2\null
3003 \ifx\null#3\null
3004 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3005 \def\@gls@checklevel{\relax}%
3006 \else
3007 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3008 \@gls@quotechar\@gls@levelchar\@gls@quotechar\@gls@levelchar}%
3009 \def\@gls@checklevel{\@gls@checklevel#3\null}%

```

```

3010   \fi
3011  \else
3012   \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3013   \@gls@quotechar\@gls@levelchar}%
3014   \ifx\null#3\null
3015     \def\@gls@checklevel{\@gls@checklevel#2!!\null}%
3016   \else
3017     \def\@gls@checklevel{\@gls@checklevel#2!#3\null}%
3018   \fi
3019  \fi
3020  \@gls@checklevel
3021 }

```

\@gls@checkactual and for ?:

```

3022 \def\@gls@checkactual#1?#2?#3\null{%
3023   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
3024   \toks@={#1}%
3025   \ifx\null#2\null
3026     \ifx\null#3\null
3027       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3028       \def\@gls@checkactual{\relax}%
3029     \else
3030       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3031       \@gls@quotechar\@gls@actualchar\@gls@quotechar\@gls@actualchar}%
3032       \def\@gls@checkactual{\@gls@checkactual#3\null}%
3033     \fi
3034   \else
3035     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3036     \@gls@quotechar\@gls@actualchar}%
3037     \ifx\null#3\null
3038       \def\@gls@checkactual{\@gls@checkactual#2??\null}%
3039     \else
3040       \def\@gls@checkactual{\@gls@checkactual#2?#3\null}%
3041     \fi
3042   \fi
3043   \@gls@checkactual
3044 }

```

\@gls@xdycheckquote As before but for use with xindy

```

3045 \def\@gls@xdycheckquote#1"#2"#3\null{%
3046   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
3047   \toks@={#1}%
3048   \ifx\null#2\null
3049     \ifx\null#3\null
3050       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
3051       \def\@gls@xdycheckquote{\relax}%
3052     \else
3053       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3054       \string"\string"}%

```

```

3055     \def\@gls@xdycheckquote{\@gls@xdycheckquote#3\null}%
3056     \fi
3057   \else
3058     \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
3059       \string\}%
3060     \ifx\null#3\null
3061       \def\@gls@xdycheckquote{\@gls@xdycheckquote#2""\null}%
3062     \else
3063       \def\@gls@xdycheckquote{\@gls@xdycheckquote#2"#3\null}%
3064     \fi
3065   \fi
3066   \@gls@xdycheckquote
3067 }

```

s@xdycheckbackslash Need to escape all backslashes for xindy. Define command that will define

```

\@gls@xdycheckbackslash
3068 \edef\def\@gls@xdycheckbackslash{%
3069   \noexpand\def\noexpand\@gls@xdycheckbackslash##1\@backslashchar
3070     ##2\@backslashchar##3\noexpand\null{%
3071     \noexpand\@gls@tmpb=\noexpand\expandafter
3072       {\noexpand\@gls@checkedmkidx}%
3073     \noexpand\toks@={##1}%
3074     \noexpand\ifx\noexpand\null##2\noexpand\null
3075       \noexpand\ifx\noexpand\null##3\noexpand\null
3076       \noexpand\edef\noexpand\@gls@checkedmkidx{%
3077         \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@}%
3078       \noexpand\def\noexpand\@gls@xdycheckbackslash{\relax}%
3079     \noexpand\else
3080       \noexpand\edef\noexpand\@gls@checkedmkidx{%
3081         \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
3082         \@backslashchar\@backslashchar\@backslashchar\@backslashchar}%
3083     \noexpand\def\noexpand\@gls@xdycheckbackslash{%
3084       \noexpand\@gls@xdycheckbackslash##3\noexpand\null}%
3085     \noexpand\fi
3086   \noexpand\else
3087     \noexpand\edef\noexpand\@gls@checkedmkidx{%
3088       \noexpand\the\noexpand\@gls@tmpb\noexpand\the\noexpand\toks@
3089       \@backslashchar\@backslashchar}%
3090   \noexpand\ifx\noexpand\null##3\noexpand\null
3091     \noexpand\def\noexpand\@gls@xdycheckbackslash{%
3092       \noexpand\@gls@xdycheckbackslash##2\@backslashchar
3093       \@backslashchar\noexpand\null}%
3094   \noexpand\else
3095     \noexpand\def\noexpand\@gls@xdycheckbackslash{%
3096       \noexpand\@gls@xdycheckbackslash##2\@backslashchar
3097       ##3\noexpand\null}%
3098   \noexpand\fi
3099   \noexpand\fi
3100   \noexpand\@gls@xdycheckbackslash

```



```

3101 }%
3102 }
    Now go ahead and define \@gls@xdycheckbackslash
3103 \def@gls@xdycheckbackslash

```

`\glsdohypertarget`

```

3104 \newlength@gls@tmplen
3105 \newcommand*{\glsdohypertarget}[2]{%
3106   \settoheight@gls@tmplen{#2}%
3107   \raisebox@gls@tmplen{\hypertarget{#1}}{#2}%
3108 }

```

`\glsdohyperlink`

```

3109 \newcommand*{\glsdohyperlink}[2]{\hyperlink{#1}{#2}}

```

`\@glslink` If `\hyperlink` is not defined `\@glslink` ignores its first argument and just does the second argument, otherwise it is equivalent to `\hyperlink`.

```

3110 \ifcsundef{hyperlink}%
3111 {%
3112   \let@glslink@secondoftwo
3113 }%
3114 {%
3115   \let@glslink@glsdohyperlink
3116 }

```

`\@glstarget` If `\hypertarget` is not defined, `\@glstarget` ignores its first argument and just does the second argument, otherwise it is equivalent to `\hypertarget`.

```

3117 \ifcsundef{hypertarget}%
3118 {%
3119   \let@glstarget@secondoftwo
3120 }%
3121 {%
3122   \let@glstarget@glsdohypertarget
3123 }

```

Glossary hyperlinks can be disabled using `\glsdisablehyper` (effect can be localised):

`\glsdisablehyper`

```

3124 \newcommand{\glsdisablehyper}{%
3125   \KV@glslink@hyperfalse
3126   \let@glslink@secondoftwo
3127   \let@glstarget@secondoftwo
3128 }

```

Glossary hyperlinks can be enabled using `\glsenablehyper` (effect can be localised):

`\glsenablehyper`

```
3129 \newcommand{\glsenablehyper}{%
3130   \KV@glslink@hypertrue
3131   \let\@glslink\glsdohyperlink
3132   \let\@glstarget\glsdohypertarget
3133 }
```

Provide some convenience commands if not already defined:

```
3134 \providecommand{\@firstofthree}[3]{#1}
3135 \providecommand{\@secondofthree}[3]{#2}
```

Syntax:

```
\gls[<options>]{<label>}[<insert text>]
```

Link to glossary entry using singular form. The link text is taken from the value of the text or first keys used when the entry was defined.

The first optional argument is a key-value list, the same as `\glslink`, the mandatory argument is the entry label. After the mandatory argument, there is another optional argument to insert extra text in the link text (the location of the inserted text is governed by `\glsdisplay` and `\glsdisplayfirst`). As with `\glslink` there is a starred version which is the same as the unstarred version but with the hyper key set to false. (Additional options can also be specified in the first optional argument.)

First determine which version is being used:

`\gls`

```
3136 \newrobustcmd*{\gls}{\@gls@hyp@opt@gls}
```

Defined the un-starred form. Need to determine if there is a final optional argument

`\@gls`

```
3137 \newcommand*{\@gls}[2] [] {%
3138   \new@ifnextchar[{\@gls@#1}{#2}]{\@gls@#1}{#2} []}%
3139 }
```

`\@gls@` Read in the final optional argument:

```
3140 \def\@gls@#1#2[#3]{%
3141   \glsdoifexists{#2}%
3142   {%
3143     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3144     \let\glsifplural\@secondoftwo
3145     \let\glscapscase\@firstofthree
3146     \let\glscustomtext\@empty
3147     \def\glsinsert{#3}%
```

Determine what the link text should be (this is stored in `\@glo@text`) Note that `\@gls@link` sets `\gls@type`.

```
3148 \def\@glo@text{\csname gls@\gls@type @entryfmt\endcsname}%
```

Call `\@gls@link`. If footnote package option has been used and the glossary type is `\acronym@type`, suppress hyperlink for first use. Likewise if the `hyperfirst=false` package option is used.

```
3149 \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3150 \ifKV@gls@link@local
3151 \glslocalunset{#2}%
3152 \else
3153 \glsunset{#2}%
3154 \fi
3155 }%
3156 }
```

`\Gls` behaves like `\gls`, but the first letter of the link text is converted to uppercase (note that if the first letter has an accent, the accented letter will need to be grouped when you define the entry). It is mainly intended for terms that start a sentence:

`\Gls`

```
3157 \newrobustcmd*{\Gls}{\@gls@hyp@opt\@Gls}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3158 \newcommand*{\@Gls}[2][ ]{%
3159 \new@ifnextchar[{\@Gls@{#1}{#2}}{\@Gls@{#1}{#2}[ ]}%
3160 }
```

`\@Gls@` Read in the final optional argument:

```
3161 \def\@Gls@#1#2[#3]{%
3162 \glsdoifexists{#2}%
3163 {%
3164 \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper
3165 \let\gls@ifplural\@secondoftwo
3166 \let\gls@caps@case\@secondofthree
3167 \let\gls@customtext\@empty
3168 \def\gls@insert{#3}%
```

Determine what the link text should be (this is stored in `\@glo@text`) Note that `\@gls@link` sets `\gls@type`.

```
3169 \def\@glo@text{\csname gls@\gls@type @entryfmt\endcsname}%
```

Call `\@gls@link` If footnote package option has been used and the glossary type is `\acronym@type`, suppress hyperlink for first use. Likewise if the `hyperfirst=false` package option is used.

```
3170 \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3171 \ifKV@glslink@local
3172 \glslocalunset{#2}%
3173 \else
3174 \glsunset{#2}%
3175 \fi
3176 }%
3177 }
```

`\GLS` behaves like `\gls`, but the link text is converted to uppercase:

`\GLS`

```
3178 \newrobustcmd*{\GLS}{\@gls@hyp@opt\@GLS}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3179 \newcommand*{\@GLS}[2] [] {%
3180 \new@ifnextchar[{\@GLS@{#1}{#2}}{\@GLS@{#1}{#2} []}%
3181 }
```

`\@GLS@` Read in the final optional argument:

```
3182 \def\@GLS@#1#2[#3]{%
3183 \glsdoifexists{#2}%
3184 {%
3185 \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3186 \let\glsifplural\@secondoftwo
3187 \let\glsapscase\@thirdofthree
3188 \let\glscustomtext\@empty
3189 \def\glsinsert{#3}%
```

Determine what the link text should be (this is stored in `\@glo@text`). Note that `\@gls@link` sets `\glstype`.

```
3190 \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
```

Call `\@gls@link` If footnote package option has been used and the glossary type is `\acronymtype`, suppress hyperlink for first use. Likewise if the `hyperfirst=false` package option is used.

```
3191 \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3192 \ifKV@glslink@local
3193 \glslocalunset{#2}%
3194 \else
3195 \glsunset{#2}%
3196 \fi
3197 }%
3198 }
```

`\glspl` behaves in the same way as `\gls` except it uses the plural form.

`\glspl`

```
3199 \newrobustcmd*{\glspl}{\@gls@hyp@opt\@glspl}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3200 \newcommand*{\@glspl}[2] [] {%
```

```
3201   \new@ifnextchar[{\@glspl@{#1}{#2}}{\@glspl@{#1}{#2} []}%
```

```
3202 }
```

`\@glspl@` Read in the final optional argument:

```
3203 \def\@glspl@#1#2[#3] {%
```

```
3204   \glsdoifexists{#2}%
```

```
3205   {%
```

```
3206     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper
```

```
3207     \let\glsifplural\@firstoftwo
```

```
3208     \let\gls caps case\@firstofthree
```

```
3209     \let\gls custom text\@empty
```

```
3210     \def\glsinsert{#3}%
```

Determine what the link text should be (this is stored in `\@glo@text`) Note that `\@gls@link` sets `\glstype`.

```
3211   \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
```

Call `\@gls@link`. If footnote package option has been used and the glossary type is `\acronymtype`, suppress hyperlink for first use. Likewise if the `hyperfirst=false` package option is used.

```
3212   \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3213   \ifKV@gls@link@local
```

```
3214     \glslocalunset{#2}%
```

```
3215   \else
```

```
3216     \glsunset{#2}%
```

```
3217   \fi
```

```
3218   }%
```

```
3219 }
```

`\Glspl` behaves in the same way as `\glspl`, except that the first letter of the link text is converted to uppercase (as with `\Gls`, if the first letter has an accent, it will need to be grouped).

`\Glspl`

```
3220 \newrobustcmd*{\Glspl}{\@gls@hyp@opt\@Glspl}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3221 \newcommand*{\@Glspl}[2] [] {%
```

```
3222   \new@ifnextchar[{\@Glspl@{#1}{#2}}{\@Glspl@{#1}{#2} []}%
```

```
3223 }
```

`\@Glspl@` Read in the final optional argument:

```
3224 \def\@Glspl@#1#2[#3]{%
3225   \glsdoifexists{#2}%
3226   {%
3227     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper

3228     \let\glsifplural\@firstoftwo
3229     \let\glscapscase\@secondofthree
3230     \let\glscustomtext\@empty
3231     \def\glsinsert{#3}%
```

Determine what the link text should be (this is stored in `\@glo@text`). This needs to be expanded so that the `\@glo@text` can be passed to `\xmakefirstuc`.

Note that `\@gls@link` sets `\glstype`.

```
3232   \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
```

Call `\@gls@link`. If footnote package option has been used and the glossary type is `\acronymtype`, suppress hyperlink for first use. Likewise if the `hyperfirst=false` package option is used.

```
3233   \@gls@link[#1]{#2}{\@glo@text}%
```

Indicate that this entry has now been used

```
3234   \ifKV@gls@link@local
3235     \glslocalunset{#2}%
3236   \else
3237     \glsunset{#2}%
3238   \fi
3239 }%
3240 }
```

`\GLSp1` behaves like `\glspl` except that all the link text is converted to uppercase.

`\GLSp1`

```
3241 \newrobustcmd*{\GLSp1}{\@gls@hyp@opt\@GLSp1}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3242 \newcommand*{\@GLSp1}[2] []{%
3243   \new@ifnextchar[{\@GLSp1@{#1}{#2}}{\@GLSp1@{#1}{#2}[]}%
3244 }
```

`\@GLSp1` Read in the final optional argument:

```
3245 \def\@GLSp1@#1#2[#3]{%
3246   \glsdoifexists{#2}%
3247   {%
3248     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper
```

```

3249 \let\glsifplural\@firstoftwo
3250 \let\glsifcaps\@thirdofthree
3251 \let\glsifcustom\@empty
3252 \def\glsinsert{#3}%

```

Determine what the link text should be (this is stored in \@glo@text) Note that \@gls@link sets \gls@type.

```

3253 \def\@glo@text{\csname gls@\gls@type @entryfmt\endcsname}%

```

Call \@gls@link. If footnote package option has been used and the glossary type is \acronymtype, suppress hyperlink for first use. Likewise if the hyperfirst=false package option is used.

```

3254 \@gls@link[#1]{#2}{\@glo@text}%

```

Indicate that this entry has now been used

```

3255 \ifKV@gls@link@local
3256 \glslocalunset{#2}%
3257 \else
3258 \glsunset{#2}%
3259 \fi
3260 }%
3261 }

```

`\glsdisp` `\glsdisp[options]{label}{text}` This is like `\gls` except that the link text is provided. This differs from `\glslink` in that it uses `\glsdisplay` or `\glsdisplayfirst` and unsets the first use flag.

First determine if we are using the starred form:

```

3262 \newrobustcmd*{\glsdisp}{\@gls@hyp@opt\@glsdisp}

```

Defined the un-starred form.

`\@glsdisp`

```

3263 \newcommand*{\@glsdisp}[3][ ]{%
3264 \glsdoifexists{#2}{%
3265 \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper
3266 \let\glsifplural\@secondoftwo
3267 \let\glsifcaps\@firstofthree
3268 \def\glsifcustom{#3}%
3269 \def\glsinsert{}}%

```

Determine what the link text should be (this is stored in \@glo@text) Note that \@gls@link sets \gls@type.

```

3270 \def\@glo@text{\csname gls@\gls@type @entryfmt\endcsname}%

```

Call \@gls@link. If footnote package option has been used and the glossary type is \acronymtype, suppress hyperlink for first use. Likewise if the hyperfirst=false package option is used.

```

3271 \@gls@link[#1]{#2}{\@glo@text}%

```

Indicate that this entry has now been used

```
3272 \ifKV@glslink@local
3273 \glsllocalunset{#2}%
3274 \else
3275 \glsunset{#2}%
3276 \fi
3277 }%
3278 }
```

`\@gls@field@link`

```
3279 \newcommand{\@gls@field@link}[3]{%
3280 \glsdoifexists{#2}%
3281 {%
3282 \let\do@gls@link@checkfirsthyper\relax
3283 \@gls@link[#1]{#2}{#3}%
3284 }%
3285 }
```

`\glsstext` behaves like `\gls` except it always uses the value given by the text key and it doesn't mark the entry as used.

`\glsstext`

```
3286 \newrobustcmd*{\glsstext}{\@gls@hyp@opt\@glsstext}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3287 \newcommand*{\@glsstext}[2][ ]{%
3288 \new@ifnextchar[{\@glsstext@{#1}{#2}}{\@glsstext@{#1}{#2}[ ]}}
```

Read in the final optional argument:

```
3289 \def\@glsstext@#1#2[#3]{%
3290 \@gls@field@link{#1}{#2}{\glsentrytext{#2}#3}%
3291 }
```

`\GLStext` behaves like `\glsstext` except the text is converted to uppercase.

`\GLStext`

```
3292 \newrobustcmd*{\GLStext}{\@gls@hyp@opt\@GLStext}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3293 \newcommand*{\@GLStext}[2][ ]{%
3294 \new@ifnextchar[{\@GLStext@{#1}{#2}}{\@GLStext@{#1}{#2}[ ]}}
```

Read in the final optional argument:

```
3295 \def\@GLStext@#1#2[#3]{%
3296 \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrytext{#2}#3}}%
3297 }
```

`\Glstext` behaves like `\glsstext` except that the first letter of the text is converted to uppercase.

`\Glstext`

```
3298 \newrobustcmd*{\Glstext}{\@gls@hyp@opt\@Glstext}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3299 \newcommand*{\@Glstext}[2] [] {%
```

```
3300   \new@ifnextchar[{\@Glstext@{#1}{#2}}{\@Glstext@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3301 \def\@Glstext@#1#2[#3] {%
```

```
3302   \@gls@field@link{#1}{#2}{\Glsentrytext{#2}#3}%
```

```
3303 }
```

`\glsfirst` behaves like `\gls` except it always uses the value given by the first key and it doesn't mark the entry as used.

`\glsfirst`

```
3304 \newrobustcmd*{\glsfirst}{\@gls@hyp@opt\@glsfirst}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3305 \newcommand*{\@glsfirst}[2] [] {%
```

```
3306   \new@ifnextchar[{\@glsfirst@{#1}{#2}}{\@glsfirst@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3307 \def\@glsfirst@#1#2[#3] {%
```

```
3308   \@gls@field@link{#1}{#2}{\Glsentryfirst{#2}#3}%
```

```
3309 }
```

`\Glsfirst` behaves like `\glsfirst` except it displays the first letter in uppercase.

`\Glsfirst`

```
3310 \newrobustcmd*{\Glsfirst}{\@gls@hyp@opt\@Glsfirst}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3311 \newcommand*{\@Glsfirst}[2] [] {%
```

```
3312   \new@ifnextchar[{\@Glsfirst@{#1}{#2}}{\@Glsfirst@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3313 \def\@Glsfirst@#1#2[#3] {%
```

```
3314   \@gls@field@link{#1}{#2}{\Glsentryfirst{#2}#3}%
```

```
3315 }
```

`\GLSfirst` behaves like `\Glsfirst` except it displays the text in uppercase.

`\GLSfirst`

```
3316 \newrobustcmd*{\GLSfirst}{\@gls@hyp@opt\@GLSfirst}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3317 \newcommand*{\@GLSfirst}[2] [] {%  
3318   \new@ifnextchar[{\@GLSfirst@{#1}{#2}}{\@GLSfirst@{#1}{#2} [] ]}
```

Read in the final optional argument:

```
3319 \def\@GLSfirst@#1#2[#3] {%  
3320   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryfirst{#2}#3}}%  
3321 }
```

`\glsplural` behaves like `\gls` except it always uses the value given by the plural key and it doesn't mark the entry as used.

`\glsplural`

```
3322 \newrobustcmd*{\glsplural}{\@gls@hyp@opt\@glsplural}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3323 \newcommand*{\@glsplural}[2] [] {%  
3324   \new@ifnextchar[{\@glsplural@{#1}{#2}}{\@glsplural@{#1}{#2} [] ]}
```

Read in the final optional argument:

```
3325 \def\@glsplural@#1#2[#3] {%  
3326   \@gls@field@link{#1}{#2}{\glsentryplural{#2}#3}}%  
3327 }
```

`\Glsplural` behaves like `\glsplural` except that the first letter is converted to uppercase.

`\Glsplural`

```
3328 \newrobustcmd*{\Glsplural}{\@gls@hyp@opt\@Glsplural}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3329 \newcommand*{\@Glsplural}[2] [] {%  
3330   \new@ifnextchar[{\@Glsplural@{#1}{#2}}{\@Glsplural@{#1}{#2} [] ]}
```

Read in the final optional argument:

```
3331 \def\@Glsplural@#1#2[#3] {%  
3332   \@gls@field@link{#1}{#2}{\Glsentryplural{#2}#3}}%  
3333 }
```

`\GLSplural` behaves like `\glsplural` except that the text is converted to uppercase.

`\GLSplural`

```
3334 \newrobustcmd*{\GLSplural}{\@gls@hyp@opt\@GLSplural}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3335 \newcommand*{\@GLSplural}[2] [] {%  
3336   \new@ifnextchar[{\@GLSplural@{#1}{#2}}{\@GLSplural@{#1}{#2} [] ]}
```

Read in the final optional argument:

```
3337 \def\@GLSplural@#1#2[#3]{%
3338   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryplural{#2}#3}}%
3339 }
```

`\glsfirstplural` behaves like `\gls` except it always uses the value given by the firstplural key and it doesn't mark the entry as used.

`\glsfirstplural`

```
3340 \newrobustcmd*{\glsfirstplural}{\@gls@hyp@opt\@glsfirstplural}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3341 \newcommand*{\@glsfirstplural}[2][ ]{%
3342   \new@ifnextchar[{\@glsfirstplural@{#1}{#2}}{\@glsfirstplural@{#1}{#2}[ ]}]
```

Read in the final optional argument:

```
3343 \def\@glsfirstplural@#1#2[#3]{%
3344   \@gls@field@link{#1}{#2}{\glsentryfirstplural{#2}#3}%
3345 }
```

`\Glsfirstplural` behaves like `\glsfirstplural` except that the first letter is converted to uppercase.

`\Glsfirstplural`

```
3346 \newrobustcmd*{\Glsfirstplural}{\@gls@hyp@opt\@Glsfirstplural}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3347 \newcommand*{\@Glsfirstplural}[2][ ]{%
3348   \new@ifnextchar[{\@Glsfirstplural@{#1}{#2}}{\@Glsfirstplural@{#1}{#2}[ ]}]
```

Read in the final optional argument:

```
3349 \def\@Glsfirstplural@#1#2[#3]{%
3350   \@gls@field@link{#1}{#2}{\Glsentryfirstplural{#2}#3}%
3351 }
```

`\GLSfirstplural` behaves like `\glsfirstplural` except that the link text is converted to uppercase.

`\GLSfirstplural`

```
3352 \newrobustcmd*{\GLSfirstplural}{\@gls@hyp@opt\@GLSfirstplural}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3353 \newcommand*{\@GLSfirstplural}[2][ ]{%
3354   \new@ifnextchar[{\@GLSfirstplural@{#1}{#2}}{\@GLSfirstplural@{#1}{#2}[ ]}]
```

Read in the final optional argument:

```
3355 \def\@GLSfirstplural@#1#2[#3]{%
3356   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryfirstplural{#2}#3}}%
3357 }
```

`\glsname` behaves like `\gls` except it always uses the value given by the name key and it doesn't mark the entry as used.

`\glsname`

```
3358 \newrobustcmd*{\glsname}{\@gls@hyp@opt\@glsname}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3359 \newcommand*{\@glsname}[2][\@glsname@#1]{\@glsname@#1}
```

```
3360 \new@ifnextchar[\@glsname@#1]{\@glsname@#1}{\@glsname@#1}
```

Read in the final optional argument:

```
3361 \def\@glsname@#1#2[#3]{\@glsname@#1#2}
```

```
3362 \@gls@field@link{#1}{#2}{\glsentryname{#2}#3}
```

```
3363 }
```

`\Glsname` behaves like `\glsname` except that the first letter is converted to uppercase.

`\Glsname`

```
3364 \newrobustcmd*{\Glsname}{\@gls@hyp@opt\@Glsname}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3365 \newcommand*{\@Glsname}[2][\@Glsname@#1]{\@Glsname@#1}
```

```
3366 \new@ifnextchar[\@Glsname@#1]{\@Glsname@#1}{\@Glsname@#1}
```

Read in the final optional argument:

```
3367 \def\@Glsname@#1#2[#3]{\@Glsname@#1#2}
```

```
3368 \@gls@field@link{#1}{#2}{\Glsentryname{#2}#3}
```

```
3369 }
```

`\GLSname` behaves like `\glsname` except that the link text is converted to uppercase.

`\GLSname`

```
3370 \newrobustcmd*{\GLSname}{\@gls@hyp@opt\@GLSname}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3371 \newcommand*{\@GLSname}[2][\@GLSname@#1]{\@GLSname@#1}
```

```
3372 \new@ifnextchar[\@GLSname@#1]{\@GLSname@#1}{\@GLSname@#1}
```

Read in the final optional argument:

```
3373 \def\@GLSname@#1#2[#3]{\@GLSname@#1#2}
```

```
3374 \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryname{#2}#3}}
```

```
3375 }
```

`\glsdesc` behaves like `\gls` except it always uses the value given by the description key and it doesn't mark the entry as used.

`\glsdesc`

```
3376 \newrobustcmd*{\glsdesc}{\@gls@hyp@opt\@glsdesc}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3377 \newcommand*{\@glsdesc}[2][ ]{%
3378   \new@ifnextchar[{\@glsdesc@{#1}{#2}}{\@glsdesc@{#1}{#2}[ ]}]}
```

Read in the final optional argument:

```
3379 \def\@glsdesc@#1#2[#3]{%
3380   \@gls@field@link{#1}{#2}{\glsentrydesc{#2}#3}%
3381 }
```

`\Glsdesc` behaves like `\glsdesc` except that the first letter is converted to uppercase.

`\Glsdesc`

```
3382 \newrobustcmd*{\Glsdesc}{\@gls@hyp@opt\@Glsdesc}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3383 \newcommand*{\@Glsdesc}[2][ ]{%
3384   \new@ifnextchar[{\@Glsdesc@{#1}{#2}}{\@Glsdesc@{#1}{#2}[ ]}]}
```

Read in the final optional argument:

```
3385 \def\@Glsdesc@#1#2[#3]{%
3386   \@gls@field@link{#1}{#2}{\Glsentrydesc{#2}#3}%
3387 }
```

`\GLSdesc` behaves like `\glsdesc` except that the link text is converted to uppercase.

`\GLSdesc`

```
3388 \newrobustcmd*{\GLSdesc}{\@gls@hyp@opt\@GLSdesc}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3389 \newcommand*{\@GLSdesc}[2][ ]{%
3390   \new@ifnextchar[{\@GLSdesc@{#1}{#2}}{\@GLSdesc@{#1}{#2}[ ]}]}
```

Read in the final optional argument:

```
3391 \def\@GLSdesc@#1#2[#3]{%
3392   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrydesc{#2}#3}}%
3393 }
```

`\glsdescplural` behaves like `\gls` except it always uses the value given by the `descriptionplural` key and it doesn't mark the entry as used.

`\glsdescplural`

```
3394 \newrobustcmd*{\glsdescplural}{\@gls@hyp@opt\@glsdescplural}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3395 \newcommand*{\@glsdescplural}[2][ ]{%
3396   \new@ifnextchar[{\@glsdescplural@{#1}{#2}}{\@glsdescplural@{#1}{#2}[ ]}]}
```

Read in the final optional argument:

```
3397 \def\@glsdescplural@#1#2[#3]{%
3398   \@gls@field@link{#1}{#2}{\glsentrydescplural{#2}#3}%
3399 }
```

`\Glsdescplural` behaves like `\glsdescplural` except that the first letter is converted to uppercase.

`\Glsdescplural`

```
3400 \newrobustcmd*{\Glsdescplural}{\@gls@hyp@opt\@Glsdescplural}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3401 \newcommand*{\@Glsdescplural}[2] []{%
3402   \new@ifnextchar[{\@Glsdescplural@{#1}{#2}}{\@Glsdescplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3403 \def\@Glsdescplural@#1#2[#3]{%
3404   \@gls@field@link{#1}{#2}{\Glsentrydescplural{#2}#3}%
3405 }
```

`\GLSdescplural` behaves like `\glsdescplural` except that the link text is converted to uppercase.

`\GLSdescplural`

```
3406 \newrobustcmd*{\GLSdescplural}{\@gls@hyp@opt\@GLSdescplural}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3407 \newcommand*{\@GLSdescplural}[2] []{%
3408   \new@ifnextchar[{\@GLSdescplural@{#1}{#2}}{\@GLSdescplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3409 \def\@GLSdescplural@#1#2[#3]{%
3410   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentrydescplural{#2}#3}}%
3411 }
```

`\glsymbol` behaves like `\gls` except it always uses the value given by the symbol key and it doesn't mark the entry as used.

`\glsymbol`

```
3412 \newrobustcmd*{\glsymbol}{\@gls@hyp@opt\@glsymbol}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3413 \newcommand*{\@glsymbol}[2] []{%
3414   \new@ifnextchar[{\@glsymbol@{#1}{#2}}{\@glsymbol@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3415 \def\@glsymbol@#1#2[#3]{%
3416   \@gls@field@link{#1}{#2}{\glsentrysymbol{#2}#3}%
3417 }
```

`\Glssymbol` behaves like `\glssymbol` except that the first letter is converted to uppercase.

`\Glssymbol`

```
3418 \newrobustcmd*{\Glssymbol}{\@gls@hyp@opt\@Glssymbol}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3419 \newcommand*{\@Glssymbol}[2] [] {%
```

```
3420   \new@ifnextchar[{\@Glssymbol@{#1}{#2}}{\@Glssymbol@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3421 \def\@Glssymbol@#1#2[#3] {%
```

```
3422   \@gls@field@link{#1}{#2}{\Glssymbol{#2}#3}%
```

```
3423 }
```

`\GLSsymbol` behaves like `\glssymbol` except that the link text is converted to uppercase.

`\GLSsymbol`

```
3424 \newrobustcmd*{\GLSsymbol}{\@gls@hyp@opt\@GLSsymbol}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3425 \newcommand*{\@GLSsymbol}[2] [] {%
```

```
3426   \new@ifnextchar[{\@GLSsymbol@{#1}{#2}}{\@GLSsymbol@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3427 \def\@GLSsymbol@#1#2[#3] {%
```

```
3428   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glssymbol{#2}#3}}%
```

```
3429 }
```

`\glssymbolplural` behaves like `\gls` except it always uses the value given by the `symbolplural` key and it doesn't mark the entry as used.

`\glssymbolplural`

```
3430 \newrobustcmd*{\glssymbolplural}{\@gls@hyp@opt\@glssymbolplural}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3431 \newcommand*{\@glssymbolplural}[2] [] {%
```

```
3432   \new@ifnextchar[{\@glssymbolplural@{#1}{#2}}{\@glssymbolplural@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3433 \def\@glssymbolplural@#1#2[#3] {%
```

```
3434   \@gls@field@link{#1}{#2}{\glssymbolplural{#2}#3}%
```

```
3435 }
```

`\Glssymbolplural` behaves like `\glssymbolplural` except that the first letter is converted to uppercase.

`\Glssymbolplural`

```
3436 \newrobustcmd*{\Glssymbolplural}{\@gls@hyp@opt\@Glssymbolplural}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3437 \newcommand*{\@Glssymbolplural}[2] [] {%
3438   \new@ifnextchar[{\@Glssymbolplural@{#1}{#2}}{\@Glssymbolplural@{#1}{#2} [] ]}
```

Read in the final optional argument:

```
3439 \def\@Glssymbolplural@#1#2[#3] {%
3440   \@gls@field@link{#1}{#2}{\Glsentrysymbolplural{#2}#3}%
3441 }
```

`\GLSsymbolplural` behaves like `\glsymbolplural` except that the link text is converted to uppercase.

`\GLSsymbolplural`

```
3442 \newrobustcmd*{\GLSsymbolplural}{\@gls@hyp@opt\@GLSsymbolplural}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3443 \newcommand*{\@GLSsymbolplural}[2] [] {%
3444   \new@ifnextchar[{\@GLSsymbolplural@{#1}{#2}}{\@GLSsymbolplural@{#1}{#2} [] ]}
```

Read in the final optional argument:

```
3445 \def\@GLSsymbolplural@#1#2[#3] {%
3446   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentrysymbolplural{#2}#3}}%
3447 }
```

`\glsuseri` behaves like `\gls` except it always uses the value given by the `user1` key and it doesn't mark the entry as used.

`\glsuseri`

```
3448 \newrobustcmd*{\glsuseri}{\@gls@hyp@opt\@glsuseri}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3449 \newcommand*{\@glsuseri}[2] [] {%
3450   \new@ifnextchar[{\@glsuseri@{#1}{#2}}{\@glsuseri@{#1}{#2} [] ]}
```

Read in the final optional argument:

```
3451 \def\@glsuseri@#1#2[#3] {%
3452   \@gls@field@link{#1}{#2}{\Glsentryuseri{#2}#3}%
3453 }
```

`\Glsuseri` behaves like `\glsuseri` except that the first letter is converted to uppercase.

`\Glsuseri`

```
3454 \newrobustcmd*{\Glsuseri}{\@gls@hyp@opt\@Glsuseri}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3455 \newcommand*{\@Glsuseri}[2] [] {%
3456   \new@ifnextchar[{\@Glsuseri@{#1}{#2}}{\@Glsuseri@{#1}{#2} [] ]}
```


Read in the final optional argument:

```
3457 \def\@Glsuseri@#1#2[#3]{%
3458   \@gls@field@link{#1}{#2}{\Glsentryuseri{#2}#3}%
3459 }
```

`\Glsuseri` behaves like `\glsuseri` except that the link text is converted to uppercase.

`\Glsuseri`

```
3460 \newrobustcmd*{\Glsuseri}{\@gls@hyp@opt\@Glsuseri}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3461 \newcommand*{\@Glsuseri}[2] [] {%
3462   \new@ifnextchar[{\@Glsuseri@{#1}{#2}}{\@Glsuseri@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3463 \def\@Glsuseri@#1#2[#3]{%
3464   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\Glsentryuseri{#2}#3}}%
3465 }
```

`\glsuserii` behaves like `\gls` except it always uses the value given by the `user2` key and it doesn't mark the entry as used.

`\glsuserii`

```
3466 \newrobustcmd*{\glsuserii}{\@gls@hyp@opt\@glsuserii}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3467 \newcommand*{\@glsuserii}[2] [] {%
3468   \new@ifnextchar[{\@glsuserii@{#1}{#2}}{\@glsuserii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3469 \def\@glsuserii@#1#2[#3]{%
3470   \@gls@field@link{#1}{#2}{\Glsentryuserii{#2}#3}%
3471 }
```

`\Glsuserii` behaves like `\glsuserii` except that the first letter is converted to uppercase.

`\Glsuserii`

```
3472 \newrobustcmd*{\Glsuserii}{\@gls@hyp@opt\@Glsuserii}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3473 \newcommand*{\@Glsuserii}[2] [] {%
3474   \new@ifnextchar[{\@Glsuserii@{#1}{#2}}{\@Glsuserii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3475 \def\@Glsuserii@#1#2[#3]{%
3476   \@gls@field@link{#1}{#2}{\Glsentryuserii{#2}#3}%
3477 }
```

`\GLSuserii` behaves like `\glsuserii` except that the link text is converted to uppercase.

`\GLSuserii`

```
3478 \newrobustcmd*{\GLSuserii}{\@gls@hyp@opt\@GLSuserii}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3479 \newcommand*{\@GLSuserii}[2] [] {%
```

```
3480 \new@ifnextchar[{\@GLSuserii@{#1}{#2}}{\@GLSuserii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3481 \def\@GLSuserii@#1#2[#3] {%
```

```
3482 \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuserii{#2}#3}}%
```

```
3483 }
```

`\glsuseriii` behaves like `\gls` except it always uses the value given by the `user3` key and it doesn't mark the entry as used.

`\glsuseriii`

```
3484 \newrobustcmd*{\glsuseriii}{\@gls@hyp@opt\@glsuseriii}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3485 \newcommand*{\@glsuseriii}[2] [] {%
```

```
3486 \new@ifnextchar[{\@glsuseriii@{#1}{#2}}{\@glsuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3487 \def\@glsuseriii@#1#2[#3] {%
```

```
3488 \@gls@field@link{#1}{#2}{\glsentryuseriii{#2}#3}}%
```

```
3489 }
```

`\Glsuseriii` behaves like `\glsuseriii` except that the first letter is converted to uppercase.

`\Glsuseriii`

```
3490 \newrobustcmd*{\Glsuseriii}{\@gls@hyp@opt\@Glsuseriii}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3491 \newcommand*{\@Glsuseriii}[2] [] {%
```

```
3492 \new@ifnextchar[{\@Glsuseriii@{#1}{#2}}{\@Glsuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3493 \def\@Glsuseriii@#1#2[#3] {%
```

```
3494 \@gls@field@link{#1}{#2}{\Glsentryuseriii{#2}#3}}%
```

```
3495 }
```

`\GLSuseriii` behaves like `\glsuseriii` except that the link text is converted to uppercase.

`\GLSuseriii`

```
3496 \newrobustcmd*{\GLSuseriii}{\@gls@hyp@opt\@GLSuseriii}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3497 \newcommand*{\@GLSuseriii}[2] [] {%
3498   \new@ifnextchar[{\@GLSuseriii@{#1}{#2}}{\@GLSuseriii@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3499 \def\@GLSuseriii@#1#2[#3]{%
3500   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuseriii{#2}#3}}%
3501 }
```

`\glsuseriv` behaves like `\gls` except it always uses the value given by the `user4` key and it doesn't mark the entry as used.

`\glsuseriv`

```
3502 \newrobustcmd*{\glsuseriv}{\@gls@hyp@opt\@glsuseriv}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3503 \newcommand*{\@glsuseriv}[2] [] {%
3504   \new@ifnextchar[{\@glsuseriv@{#1}{#2}}{\@glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3505 \def\@glsuseriv@#1#2[#3]{%
3506   \@gls@field@link{#1}{#2}{\glsentryuseriv{#2}#3}%
3507 }
```

`\Glsuseriv` behaves like `\glsuseriv` except that the first letter is converted to uppercase.

`\Glsuseriv`

```
3508 \newrobustcmd*{\Glsuseriv}{\@gls@hyp@opt\@Glsuseriv}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3509 \newcommand*{\@Glsuseriv}[2] [] {%
3510   \new@ifnextchar[{\@Glsuseriv@{#1}{#2}}{\@Glsuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3511 \def\@Glsuseriv@#1#2[#3]{%
3512   \@gls@field@link{#1}{#2}{\Glsentryuseriv{#2}#3}%
3513 }
```

`\GLSuseriv` behaves like `\glsuseriv` except that the link text is converted to uppercase.

`\GLSuseriv`

```
3514 \newrobustcmd*{\GLSuseriv}{\@gls@hyp@opt\@GLSuseriv}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3515 \newcommand*{\@GLSuseriv}[2] [] {%
3516   \new@ifnextchar[{\@GLSuseriv@{#1}{#2}}{\@GLSuseriv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3517 \def\@GLSuseriv@#1#2[#3]{%
3518   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuseriv{#2}#3}}%
3519 }
```

`\glsuserv` behaves like `\gls` except it always uses the value given by the `user5` key and it doesn't mark the entry as used.

`\glsuserv`

```
3520 \newrobustcmd*{\glsuserv}{\@gls@hyp@opt\@glsuserv}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3521 \newcommand*{\@glsuserv}[2] [] {%
3522   \new@ifnextchar[{\@glsuserv@{#1}{#2}}{\@glsuserv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3523 \def\@glsuserv@#1#2[#3]{%
3524   \@gls@field@link{#1}{#2}{\glsentryuserv{#2}#3}%
3525 }
```

`\Glsuserv` behaves like `\glsuserv` except that the first letter is converted to uppercase.

`\Glsuserv`

```
3526 \newrobustcmd*{\Glsuserv}{\@gls@hyp@opt\@Glsuserv}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3527 \newcommand*{\@Glsuserv}[2] [] {%
3528   \new@ifnextchar[{\@Glsuserv@{#1}{#2}}{\@Glsuserv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3529 \def\@Glsuserv@#1#2[#3]{%
3530   \@gls@field@link{#1}{#2}{\Glsentryuserv{#2}#3}%
3531 }
```

`\GLSuserv` behaves like `\glsuserv` except that the link text is converted to uppercase.

`\GLSuserv`

```
3532 \newrobustcmd*{\GLSuserv}{\@gls@hyp@opt\@GLSuserv}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3533 \newcommand*{\@GLSuserv}[2] [] {%
3534   \new@ifnextchar[{\@GLSuserv@{#1}{#2}}{\@GLSuserv@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3535 \def\@GLSuserv@#1#2[#3]{%
3536   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuserv{#2}#3}}%
3537 }
```

`\glsuservi` behaves like `\gls` except it always uses the value given by the `user6` key and it doesn't mark the entry as used.

`\glsuservi`

```
3538 \newrobustcmd*{\glsuservi}{\@gls@hyp@opt\@glsuservi}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3539 \newcommand*{\@glsuservi}[2] []{%
```

```
3540   \new@ifnextchar[{\@glsuservi@{#1}{#2}}{\@glsuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3541 \def\@glsuservi@#1#2[#3]{%
```

```
3542   \@gls@field@link{#1}{#2}{\glsentryuservi{#2}#3}%
```

```
3543 }
```

`\Glsuservi` behaves like `\glsuservi` except that the first letter is converted to uppercase.

`\Glsuservi`

```
3544 \newrobustcmd*{\Glsuservi}{\@gls@hyp@opt\@Glsuservi}
```

Defined the un-starred form. Need to determine if there is a final optional argument

```
3545 \newcommand*{\@Glsuservi}[2] []{%
```

```
3546   \new@ifnextchar[{\@Glsuservi@{#1}{#2}}{\@Glsuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3547 \def\@Glsuservi@#1#2[#3]{%
```

```
3548   \@gls@field@link{#1}{#2}{\Glsentryuservi{#2}#3}%
```

```
3549 }
```

`\GLSuservi` behaves like `\glsuservi` except that the link text is converted to uppercase.

`\GLSuservi`

```
3550 \newrobustcmd*{\GLSuservi}{\@gls@hyp@opt\@GLSuservi}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3551 \newcommand*{\@GLSuservi}[2] []{%
```

```
3552   \new@ifnextchar[{\@GLSuservi@{#1}{#2}}{\@GLSuservi@{#1}{#2} []}]}
```

Read in the final optional argument:

```
3553 \def\@GLSuservi@#1#2[#3]{%
```

```
3554   \@gls@field@link{#1}{#2}{\mfirstucMakeUppercase{\glsentryuservi{#2}#3}}%
```

```
3555 }
```

Now deal with acronym related keys. First the short form:

`\acrshort`

```
3556 \newrobustcmd*{\acrshort}{\@gls@hyp@opt\@ns@acrshort}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3557 \newcommand*{\ns@acrshort}[2] [] {%
3558   \new@ifnextchar[{\@acrshort{#1}{#2}}{\@acrshort{#1}{#2} []}]%
3559 }
```

Read in the final optional argument:

```
3560 \def\@acrshort#1#2[#3] {%
3561   \glsdoifexists{#2}%
3562   {%
3563     \let\do@gls@link@checkfirsthyper\relax

3564     \let\glsifplural\@secondoftwo
3565     \let\glsapscase\@firstofthree
3566     \let\glsinsert\@empty
3567     \def\glscustomtext{%
3568       \acronymfont{\glsentryshort{#2}}#3%
3569     }%
```

Call \@gls@link Note that \@gls@link sets \glsstyle.

```
3570   \@gls@link[#1]{#2}{\csname gls@glstyle @entryfmt\endcsname}%
3571   }%
3572 }
```

\Acrshort

```
3573 \newrobustcmd*{\Acrshort}{\@gls@hyp@opt\ns@Acrshort}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3574 \newcommand*{\ns@Acrshort}[2] [] {%
3575   \new@ifnextchar[{\@Acrshort{#1}{#2}}{\@Acrshort{#1}{#2} []}]%
3576 }
```

Read in the final optional argument:

```
3577 \def\@Acrshort#1#2[#3] {%
3578   \glsdoifexists{#2}%
3579   {%
3580     \let\do@gls@link@checkfirsthyper\relax

3581     \def\glslabel{#2}%
3582     \let\glsifplural\@secondoftwo
3583     \let\glsapscase\@secondofthree
3584     \let\glsinsert\@empty
3585     \def\glscustomtext{%
3586       \Glsentryshort{#2}}#3%
3587     }%
```

Call \@gls@link Note that \@gls@link sets \glsstyle.

```
3588   \@gls@link[#1]{#2}{\csname gls@glstyle @entryfmt\endcsname}%
3589   }%
3590 }
```

\ACRshort

```
3591 \newrobustcmd*{\ACRshort}{\@gls@hyp@opt\ns@ACRshort}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3592 \newcommand*{\ns@ACRshort}[2] [] {%
3593   \new@ifnextchar[{\@ACRshort{#1}{#2}}{\@ACRshort{#1}{#2} []}]%
3594 }
```

Read in the final optional argument:

```
3595 \def\@ACRshort#1#2[#3] {%
3596   \glsdoifexists{#2}%
3597   {%
3598     \let\do@gls@link@checkfirsthyper\relax
3599
3600     \def\glslabel{#2}%
3601     \let\glsifplural\@secondoftwo
3602     \let\gls caps case\@thirdofthree
3603     \let\glsinsert\@empty
3604     \def\gls custom text{%
3605       \mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}%
3606     }%
3607   }
```

Call \@gls@link Note that \@gls@link sets \gls type.

```
3606   \@gls@link[#1]{#2}{\csname gls@gls type @entryfmt\endcsname}%
3607 }%
3608 }
```

Short plural:

\acrshortpl

```
3609 \newrobustcmd*{\acrshortpl}{\@gls@hyp@opt\ns@acrshortpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3610 \newcommand*{\ns@acrshortpl}[2] [] {%
3611   \new@ifnextchar[{\@acrshortpl{#1}{#2}}{\@acrshortpl{#1}{#2} []}]%
3612 }
```

Read in the final optional argument:

```
3613 \def\@acrshortpl#1#2[#3] {%
3614   \glsdoifexists{#2}%
3615   {%
3616     \let\do@gls@link@checkfirsthyper\relax
3617
3618     \def\glslabel{#2}%
3619     \let\glsifplural\@firstoftwo
3620     \let\gls caps case\@firstofthree
3621     \let\glsinsert\@empty
3622     \def\gls custom text{%
3623       \acronymfont{\glsentryshortpl{#2}}#3%
3624     }%
3625   }
```

Call `\@gls@link` Note that `\@gls@link` sets `\glstype`.

```
3624 \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%  
3625 }%  
3626 }
```

`\Acrshortpl`

```
3627 \newrobustcmd*{\Acrshortpl}{\@gls@hyp@opt\ns@Acrshortpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3628 \newcommand*{\ns@Acrshortpl}[2] [] {%  
3629 \new@ifnextchar[{\@Acrshortpl{#1}{#2}}{\@Acrshortpl{#1}{#2} []}%  
3630 }
```

Read in the final optional argument:

```
3631 \def\@Acrshortpl#1#2[#3] {%  
3632 \glsdoifexists{#2}%  
3633 {%  
3634 \let\do@gls@link@checkfirsthyper\relax  
  
3635 \def\glslabel{#2}%  
3636 \let\glsifplural\@firstoftwo  
3637 \let\glsapscase\@secondofthree  
3638 \let\glsinsert\@empty  
3639 \def\glscustomtext{%  
3640 \acronymfont{\Glsentryshortpl{#2}}#3%  
3641 }%
```

Call `\@gls@link` Note that `\@gls@link` sets `\glstype`.

```
3642 \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%  
3643 }%  
3644 }
```

`\ACRshortpl`

```
3645 \newrobustcmd*{\ACRshortpl}{\@gls@hyp@opt\ns@ACRshortpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3646 \newcommand*{\ns@ACRshortpl}[2] [] {%  
3647 \new@ifnextchar[{\@ACRshortpl{#1}{#2}}{\@ACRshortpl{#1}{#2} []}%  
3648 }
```

Read in the final optional argument:

```
3649 \def\@ACRshortpl#1#2[#3] {%  
3650 \glsdoifexists{#2}%  
3651 {%  
3652 \let\do@gls@link@checkfirsthyper\relax
```



```

3653 \def\glslabel{#2}%
3654 \let\glsifplural\@firstoftwo
3655 \let\glsifcaps\@thirdofthree
3656 \let\glsinsert\@empty
3657 \def\glscustomtext{%
3658 \mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}%
3659 }%

```

Call `\@gls@link` Note that `\@gls@link` sets `\gls@type`.

```

3660 \@gls@link[#1]{#2}{\csname gls@\gls@type @entryfmt\endcsname}%
3661 }%
3662 }

```

`\acrlong`

```

3663 \newrobustcmd*{\acrlong}{\@gls@hyp@opt\@ns@acrlong}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3664 \newcommand*{\ns@acrlong}[2] [] {%
3665 \new@ifnextchar[{\@acrlong{#1}{#2}}{\@acrlong{#1}{#2} []}%
3666 }

```

Read in the final optional argument:

```

3667 \def\@acrlong#1#2[#3] {%
3668 \glsdoifexists{#2}%
3669 {%
3670 \let\do@gls@link@checkfirsthyper\relax

```

```

3671 \def\glslabel{#2}%
3672 \let\glsifplural\@secondoftwo
3673 \let\glsifcaps\@firstofthree
3674 \let\glsinsert\@empty

```

Bug fix v4.02 removed `\acronymfont` from `\glscustomtext` (`\acronymfont` only designed for short form).

```

3675 \def\glscustomtext{%
3676 \glsentrylong{#2}#3%
3677 }%

```

Call `\@gls@link` Note that `\@gls@link` sets `\gls@type`.

```

3678 \@gls@link[#1]{#2}{\csname gls@\gls@type @entryfmt\endcsname}%
3679 }%
3680 }

```

`\Acrlong`

```

3681 \newrobustcmd*{\Acrlong}{\@gls@hyp@opt\@ns@Acrlong}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3682 \newcommand*{\ns@Acrlong}[2] [] {%

```

```

3683 \new@ifnextchar[{\@Acrlong{#1}{#2}}{\@Acrlong{#1}{#2} []}]%
3684 }

```

Read in the final optional argument:

```

3685 \def\@Acrlong#1#2[#3]{%
3686 \glsdoifexists{#2}%
3687 {%
3688 \let\do@gls@link@checkfirsthyper\relax

3689 \def\glslabel{#2}%
3690 \let\glsifplural\@secondoftwo
3691 \let\gls caps case\@secondofthree
3692 \let\glsinsert\@empty

```

Bug fix v4.02 removed `\acronymfont` from `\gls customtext` (`\acronymfont` only designed for short form).

```

3693 \def\gls customtext{%
3694 \Glsentrylong{#2}#3%
3695 }%

```

Call `\@gls@link`. Note that `\@gls@link` sets `\gls type`.

```

3696 \@gls@link[#1]{#2}{\csname gls@\gls type @entryfmt\endcsname}%
3697 }%
3698 }

```

`\ACRlong`

```

3699 \newrobustcmd*{\ACRlong}{\@gls@hyp@opt\@ns@ACRlong}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3700 \newcommand*{\ns@ACRlong}[2] [] {%
3701 \new@ifnextchar[{\@ACRlong{#1}{#2}}{\@ACRlong{#1}{#2} []}]%
3702 }

```

Read in the final optional argument:

```

3703 \def\@ACRlong#1#2[#3]{%
3704 \glsdoifexists{#2}%
3705 {%
3706 \let\do@gls@link@checkfirsthyper\relax

3707 \def\glslabel{#2}%
3708 \let\glsifplural\@secondoftwo
3709 \let\gls caps case\@thirdofthree
3710 \let\glsinsert\@empty

```

Bug fix v4.02 removed `\acronymfont` from `\gls customtext` (`\acronymfont` only designed for short form).

```

3711 \def\gls customtext{%
3712 \mfirstucMakeUppercase{\glsentrylong{#2}#3}%
3713 }%

```

Call `\@gls@link`. Note that `\@gls@link` sets `\glstype`.

```
3714 \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%  
3715 }%  
3716 }
```

Short plural:

`\acrlongpl`

```
3717 \newrobustcmd*{\acrlongpl}{\@gls@hyp@opt\ns@acrlongpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3718 \newcommand*{\ns@acrlongpl}[2] [] {%  
3719 \new@ifnextchar[{\@acrlongpl{#1}{#2}}{\@acrlongpl{#1}{#2} []}%  
3720 }
```

Read in the final optional argument:

```
3721 \def\@acrlongpl#1#2[#3] {%  
3722 \glsdoifexists{#2}%  
3723 {%  
3724 \let\do@gls@link@checkfirsthyper\relax  
  
3725 \def\glslabel{#2}%  
3726 \let\glsifplural\@firstoftwo  
3727 \let\glsapscase\@firstofthree  
3728 \let\glsinsert\@empty
```

Bug fix v4.02 removed `\acronymfont` from `\glscustomtext` (`\acronymfont` only designed for short form).

```
3729 \def\glscustomtext{%  
3730 \glsentrylongpl{#2}#3%  
3731 }%
```

Call `\@gls@link`. Note that `\@gls@link` sets `\glstype`.

```
3732 \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%  
3733 }%  
3734 }
```

`\Acrlongpl`

```
3735 \newrobustcmd*{\Acrlongpl}{\@gls@hyp@opt\ns@Acrlongpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
3736 \newcommand*{\ns@Acrlongpl}[2] [] {%  
3737 \new@ifnextchar[{\@Acrlongpl{#1}{#2}}{\@Acrlongpl{#1}{#2} []}%  
3738 }
```

Read in the final optional argument:

```
3739 \def\@Acrlongpl#1#2[#3] {%  
3740 \glsdoifexists{#2}%  
3741 {%  
3742 \let\do@gls@link@checkfirsthyper\relax
```

```

3743 \def\glslabel{#2}%
3744 \let\glsifplural\@firstoftwo
3745 \let\glsifcaps\@secondofthree
3746 \let\glsinsert\@empty

```

Bug fix v4.02 removed `\acronymfont` from `\glscustomtext` (`\acronymfont` only designed for short form).

```

3747 \def\glscustomtext{%
3748 \Glsentrylongpl{#2}#3%
3749 }%

```

Call `\@gls@link`. Note that `\@gls@link` sets `\gls@type`.

```

3750 \@gls@link[#1]{#2}{\csname gls@\gls@type @entryfmt\endcsname}%
3751 }%
3752 }

```

`\ACRlongpl`

```

3753 \newrobustcmd*{\ACRlongpl}{\@gls@hyp@opt\@ns@ACRlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

3754 \newcommand*{\ns@ACRlongpl}[2] []{%
3755 \new@ifnextchar[{\@ACRlongpl{#1}{#2}}{\@ACRlongpl{#1}{#2} []}%
3756 }

```

Read in the final optional argument:

```

3757 \def\@ACRlongpl#1#2[#3]{%
3758 \glsdoifexists{#2}%
3759 {%
3760 \let\do@gls@link@checkfirsthyper\relax

3761 \def\glslabel{#2}%
3762 \let\glsifplural\@firstoftwo
3763 \let\glsifcaps\@thirdofthree
3764 \let\glsinsert\@empty

```

Bug fix v4.02 removed `\acronymfont` from `\glscustomtext` (`\acronymfont` only designed for short form).

```

3765 \def\glscustomtext{%
3766 \mfirstucMakeUppercase{\glsentrylongpl{#2}#3}%
3767 }%

```

Call `\@gls@link`. Note that `\@gls@link` sets `\gls@type`.

```

3768 \@gls@link[#1]{#2}{\csname gls@\gls@type @entryfmt\endcsname}%
3769 }%
3770 }

```

1.10.2 Displaying entry details without adding information to the glossary

These commands merely display entry information without adding entries in the associated file or having hyperlinks.

`\@gls@entry@field` Generic version.

```
\@gls@entry@field{<label>}{<field>}
```

```
3771 \newcommand*{\@gls@entry@field}[2]{%
3772   \csname glo@glstdetoklabel{#1}@#2\endcsname
3773 }
```

`\glsletentryfield` `\glsletentryfield{<cs>}{<label>}{<field>}`

```
3774 \newcommand*{\glsletentryfield}[3]{%
3775   \letcs{#1}{glo@glstdetoklabel{#2}@#3}%
3776 }
```

`\@Gls@entry@field` Generic first letter uppercase version.

```
\@Gls@entry@field{<label>}{<field>}
```

```
3777 \newcommand*{\@Gls@entry@field}[2]{%
3778   \letcs@glo@text{glo@glstdetoklabel{#1}@#2}%
3779   \ifdef@glo@text
3780     {%
3781       \xmakefirstuc@glo@text}%
3782     }%
3783     {%
3784       \PackageError{glossaries}{Either glossary entry
3785         ‘\glstdetoklabel{#1}’ doesn’t exist or the field ‘#2’
3786         doesn’t exist}{Check you have correctly spelt the entry
3787         label and the field name}%
3788     }%
3789 }
```

Get the entry name (as specified by the name key when the entry was defined). The argument is the label associated with the entry. Note that unless you used `name=false` in the `sanitize` package option you may get unexpected results if the name key contains any commands.

`\glsentryname`

```
3790 \newcommand*{\glsentryname}[1]{\@gls@entry@field{#1}{name}}
```

```

\Glsentryname
3791 \newrobustcmd*{\Glsentryname}[1]{%
3792   \@Gls@entryname{#1}%
3793 }

```

```

\@Gls@entryname This is a workaround in the event that the user defies the warning in the manual
about not using \Glsname or \Glsentryname with acronyms. First the default
behaviour:
3794 \newcommand*{\@Gls@entryname}[1]{%
3795   \@Gls@entry@field{#1}{name}%
3796 }

```

```

\@Gls@acentryname Now the behaviour when \setacronymstyle is used:
3797 \newcommand*{\@Gls@acentryname}[1]{%
3798   \ifglshaslong{#1}%
3799   {%
3800     \letcs@glo@text@glo@glsdetoklabel{#1}@name}%
3801     \expandafter@glsg@getbody@glo@text{}@nil
3802     \expandafter@ifx@gls@body@glsentrylong\relax
3803     \expandafter\Glsentrylong@gls@rest
3804   \else
3805     \expandafter@ifx@gls@body@glsentryshort\relax
3806     \expandafter\Glsentryshort@gls@rest
3807   \else
3808     \expandafter@ifx@gls@body\acronymfont\relax

```

Temporarily make `\glsentryshort` behave like `\Glsentryshort`. (This is on the assumption that the argument of `\acronymfont` is `\glsentryshort{<label>}`, as that's the behaviour of the predefined acronym styles.) This is scoped to localise the effect of the assignment.

```

3809     {%
3810       \let\glsentryshort\Glsentryshort
3811       \@glo@text
3812     }%
3813   \else
3814     \xmakefirstuc{\@glo@text}%
3815   \fi
3816 \fi
3817 \fi
3818 }%
3819 {%

```

Not an acronym

```

3820   \@Gls@entry@field{#1}{name}%
3821 }%
3822 }

```

Get the entry description (as specified by the description when the entry was defined). The argument is the label associated with the entry. Note that un-

less you used `description=false` in the `sanitize` package option you may get unexpected results if the `description` key contained any commands.

`\glsentrydesc`

```
3823 \newcommand*{\glsentrydesc}[1]{\@gls@entry@field{#1}{desc}}
```

`\Glsentrydesc`

```
3824 \newrobustcmd*{\Glsentrydesc}[1]{%
3825   \@Gls@entry@field{#1}{desc}%
3826 }
```

Plural form:

`\glsentrydescplural`

```
3827 \newcommand*{\glsentrydescplural}[1]{%
3828   \@gls@entry@field{#1}{descplural}%
3829 }
```

`\Glsentrydescplural`

```
3830 \newrobustcmd*{\Glsentrydescplural}[1]{%
3831   \@Gls@entry@field{#1}{descplural}%
3832 }
```

Get the entry text, as specified by the `text` key when the entry was defined. The argument is the label associated with the entry:

`\glsentrytext`

```
3833 \newcommand*{\glsentrytext}[1]{\@gls@entry@field{#1}{text}}
```

`\Glsentrytext`

```
3834 \newrobustcmd*{\Glsentrytext}[1]{%
3835   \@Gls@entry@field{#1}{text}%
3836 }
```

Get the plural form:

`\glsentryplural`

```
3837 \newcommand*{\glsentryplural}[1]{%
3838   \@gls@entry@field{#1}{plural}%
3839 }
```

`\Glsentryplural`

```
3840 \newrobustcmd*{\Glsentryplural}[1]{%
3841   \@Gls@entry@field{#1}{plural}%
3842 }
```

Get the symbol associated with this entry. The argument is the label associated with the entry.

`\glsentrysymbol`

```
3843 \newcommand*{\glsentrysymbol}[1]{%
3844   \@gls@entry@field{#1}{symbol}%
3845 }
```

`\Glsentrysymbol`

```
3846 \newrobustcmd*{\Glsentrysymbol}[1]{%
3847   \@Gls@entry@field{#1}{symbol}%
3848 }
```

Plural form:

`lentrysymbolplural`

```
3849 \newcommand*{\glsentrysymbolplural}[1]{%
3850   \@gls@entry@field{#1}{symbolplural}%
3851 }
```

`lentrysymbolplural`

```
3852 \newrobustcmd*{\Glsentrysymbolplural}[1]{%
3853   \@Gls@entry@field{#1}{symbolplural}%
3854 }
```

Get the entry text to be used when the entry is first used in the document (as specified by the first key when the entry was defined).

`\glsentryfirst`

```
3855 \newcommand*{\glsentryfirst}[1]{%
3856   \@gls@entry@field{#1}{first}%
3857 }
```

`\Glsentryfirst`

```
3858 \newrobustcmd*{\Glsentryfirst}[1]{%
3859   \@Gls@entry@field{#1}{first}%
3860 }
```

Get the plural form (as specified by the firstplural key when the entry was defined).

`glsentryfirstplural`

```
3861 \newcommand*{\glsentryfirstplural}[1]{%
3862   \@gls@entry@field{#1}{firstpl}%
3863 }
```

`Glsentryfirstplural`

```
3864 \newrobustcmd*{\Glsentryfirstplural}[1]{%
3865   \@Gls@entry@field{#1}{firstpl}%
3866 }
```


Display the glossary type with which this entry is associated (as specified by the type key used when the entry was defined)

`\glsentrytype`

```
3867 \newcommand*{\glsentrytype}[1]{\@gls@entry@field{#1}{type}}
```

Display the sort text used for this entry. Note that the sort key is sanitize, so unexpected results may occur if the sort key contained commands.

`\glsentrysort`

```
3868 \newcommand*{\glsentrysort}[1]{%
3869   \@gls@entry@field{#1}{sort}%
3870 }
```

`\glsentryuseri` Get the first user key (as specified by the user1 when the entry was defined). The argument is the label associated with the entry.

```
3871 \newcommand*{\glsentryuseri}[1]{%
3872   \@gls@entry@field{#1}{useri}%
3873 }
```

`\Glsentryuseri`

```
3874 \newrobustcmd*{\Glsentryuseri}[1]{%
3875   \@Gls@entry@field{#1}{useri}%
3876 }
```

`\glsentryuserii` Get the second user key (as specified by the user2 when the entry was defined). The argument is the label associated with the entry.

```
3877 \newcommand*{\glsentryuserii}[1]{%
3878   \@gls@entry@field{#1}{userii}%
3879 }
```

`\Glsentryuserii`

```
3880 \newrobustcmd*{\Glsentryuserii}[1]{%
3881   \@Gls@entry@field{#1}{userii}%
3882 }
```

`\glsentryuseriii` Get the third user key (as specified by the user3 when the entry was defined). The argument is the label associated with the entry.

```
3883 \newcommand*{\glsentryuseriii}[1]{%
3884   \@gls@entry@field{#1}{useriii}%
3885 }
```

`\Glsentryuseriii`

```
3886 \newrobustcmd*{\Glsentryuseriii}[1]{%
3887   \@Gls@entry@field{#1}{useriii}%
3888 }
```

`\glentryuseriv` Get the fourth user key (as specified by the user4 when the entry was defined).
The argument is the label associated with the entry.

```
3889 \newcommand*{\glentryuseriv}[1]{%  
3890   \@gls@entry@field{#1}{useriv}%  
3891 }
```

`\Glsentryuseriv`

```
3892 \newrobustcmd*{\Glsentryuseriv}[1]{%  
3893   \@Gls@entry@field{#1}{useriv}%  
3894 }
```

`\glentryuseriv` Get the fifth user key (as specified by the user5 when the entry was defined).
The argument is the label associated with the entry.

```
3895 \newcommand*{\glentryuseriv}[1]{%  
3896   \@gls@entry@field{#1}{useriv}%  
3897 }
```

`\Glsentryuseriv`

```
3898 \newrobustcmd*{\Glsentryuseriv}[1]{%  
3899   \@Gls@entry@field{#1}{useriv}%  
3900 }
```

`\glentryuseriv` Get the sixth user key (as specified by the user6 when the entry was defined).
The argument is the label associated with the entry.

```
3901 \newcommand*{\glentryuseriv}[1]{%  
3902   \@gls@entry@field{#1}{useriv}%  
3903 }
```

`\Glsentryuseriv`

```
3904 \newrobustcmd*{\Glsentryuseriv}[1]{%  
3905   \@Gls@entry@field{#1}{useriv}%  
3906 }
```

`\glentryshort` Get the short key (as specified by the short the entry was defined). The argument is the label associated with the entry.

```
3907 \newcommand*{\glentryshort}[1]{\@gls@entry@field{#1}{short}}
```

`\Glsentryshort`

```
3908 \newrobustcmd*{\Glsentryshort}[1]{%  
3909   \@Gls@entry@field{#1}{short}%  
3910 }
```

`\glentryshortpl` Get the short plural key (as specified by the shortplural the entry was defined).
The argument is the label associated with the entry.

```
3911 \newcommand*{\glentryshortpl}[1]{\@gls@entry@field{#1}{shortpl}}
```

`\Glsentryshortpl`

```
3912 \newrobustcmd*{\Glsentryshortpl}[1]{%
3913   \@Gls@entry@field{#1}{shortpl}}%
3914 }
```

`\glsentrylong` Get the long key (as specified by the long the entry was defined). The argument is the label associated with the entry.

```
3915 \newcommand*{\glsentrylong}[1]{\@Gls@entry@field{#1}{long}}
```

`\Glsentrylong`

```
3916 \newrobustcmd*{\Glsentrylong}[1]{%
3917   \@Gls@entry@field{#1}{long}}%
3918 }
```

`\glsentrylongpl` Get the long plural key (as specified by the longplural the entry was defined). The argument is the label associated with the entry.

```
3919 \newcommand*{\glsentrylongpl}[1]{\@Gls@entry@field{#1}{longpl}}
```

`\Glsentrylongpl`

```
3920 \newrobustcmd*{\Glsentrylongpl}[1]{%
3921   \@Gls@entry@field{#1}{longpl}}%
3922 }
```

Short cut macros to access full form:

`\glsentryfull`

```
3923 \newcommand*{\glsentryfull}[1]{%
3924   \acrfullformat{\glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
3925 }
```

`\Glsentryfull`

```
3926 \newrobustcmd*{\Glsentryfull}[1]{%
3927   \acrfullformat{\Glsentrylong{#1}}{\acronymfont{\glsentryshort{#1}}}%
3928 }
```

`\glsentryfullpl`

```
3929 \newcommand*{\glsentryfullpl}[1]{%
3930   \acrfullformat{\glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
3931 }
```

`\Glsentryfullpl`

```
3932 \newrobustcmd*{\Glsentryfullpl}[1]{%
3933   \acrfullformat{\Glsentrylongpl{#1}}{\acronymfont{\glsentryshortpl{#1}}}%
3934 }
```

`\glsentrynumberlist` Displays the number list as is.

```
3935 \newcommand*{\glsentrynumberlist}[1]{%
3936   \glsdoifexists{#1}%
3937   {%
3938     \@gls@entry@field{#1}{numberlist}%
3939   }%
3940 }
```

`\glsdisplaynumberlist` Formats the number list for the given entry label. Doesn't work with hyperref.

```
3941 \@ifpackageloaded{hyperref} {%
3942   \newcommand*{\glsdisplaynumberlist}[1]{%
3943     \GlossariesWarning
3944     {%
3945       \string\glsdisplaynumberlist\space
3946       doesn't work with hyperref.^^JUsing
3947       \string\glsentrynumberlist\space instead%
3948     }%
3949     \glsentrynumberlist{#1}%
3950   }%
3951 }%
3952 {%
3953   \newcommand*{\glsdisplaynumberlist}[1]{%
3954     \glsdoifexists{#1}%
3955     {%
3956       \bgroup
3957
3958       \edef\@glo@label{\glsdetoklabel{#1}}%
3959       \let\@org@glsnumberformat\glsnumberformat
3960       \def\glsnumberformat##1{##1}%
3961       \protected@edef\the@numberlist{%
3962         \csname glo@\@glo@label @numberlist\endcsname}%
3963       \def\@gls@numlist@sep{}%
3964       \def\@gls@numlist@nextsep{}%
3965       \def\@gls@numlist@lastsep{}%
3966       \def\@gls@thislist{}%
3967       \def\@gls@donext@def{}%
3968       \renewcommand\do[1]{%
3969         \protected@edef\@gls@thislist{%
3970           \@gls@thislist
3971           \noexpand\@gls@numlist@sep
3972           ##1%
3973         }%
3974         \let\@gls@numlist@sep\@gls@numlist@nextsep
3975         \def\@gls@numlist@nextsep{\glsnumlistsep}%
3976         \@gls@donext@def
3977         \def\@gls@donext@def{%
3978           \def\@gls@numlist@lastsep{\glsnumlistlastsep}%
3979         }%
3980       }%
3981     }%
3982   }%
3983 }
```

```

3980     \expandafter \glsnumlistparser \expandafter{\the@numberlist}%
3981     \let\@gls@numlist@sep\@gls@numlist@lastsep
3982     \@gls@thislist
3983     \egroup
3984   }%
3985 }
3986 }

```

`\glsnumlistsep`

```
3987 \newcommand*{\glsnumlistsep}{, }
```

`\glsnumlistlastsep`

```
3988 \newcommand*{\glsnumlistlastsep}{ \& }
```

`\glshyperlink` Provide a hyperlink to a glossary entry without adding information to the glossary file. The entry needs to be added using a command like `\glslink` or `\glsadd` to ensure that the target is defined. The first (optional) argument specifies the link text. The entry name is used by default. The second argument is the entry label.

```

3989 \newcommand*{\glshyperlink}[2][\glsentrytext{\@glo@label}]{%
3990   \def\@glo@label{#2}%
3991   \@glslink{\glo@linkprefix\glsdetoklabel{#2}}{#1}}

```

1.11 Adding an entry to the glossary without generating text

The following keys are provided for `\glsadd` and `\glsaddall`:

```
3992 \define@key{glossadd}{counter}{\def\@gls@counter{#1}}
```

```
3993 \define@key{glossadd}{format}{\def\@glsnumberformat{#1}}
```

This key is only used by `\glsaddall`:

```
3994 \define@key{glossadd}{types}{\def\@glo@type{#1}}
```

```
\glsadd[<options>]{<label>}
```

Add a term to the glossary without generating any link text. The optional argument indicates which counter to use, and how to format it (using a key-value list) the second argument is the entry label. Note that *<options>* only has two keys: counter and format (the types key will be ignored).

`\glsadd`

```

3995 \newrobustcmd*{\glsadd}[2][ ]{%
3996   \glsdoifexists{#2}%
3997   {%
3998     \def\@glsnumberformat{glsnumberformat}%
3999     \edef\@gls@counter{\csname glo@glsdetoklabel{#2}@counter\endcsname}%
4000     \setkeys{glossadd}{#1}%

```

Store the entry's counter in `\theglentrycounter`

```
4001 \gls@saveentrycounter
4002 \do@wrglossary{#2}%
4003 }%
4004 }
```

`\glsaddall` [*option list*]

Add all terms defined for the listed glossaries (without displaying any text). If types key is omitted, apply to all glossary types.

`\glsaddall`

```
4005 \newrobustcmd*{\glsaddall}[1] [] {%
4006 \edef\glo@type{\glo@types}%
4007 \setkeys{glossadd}{#1}%
4008 \forallglsentries[\glo@type]{\glo@entry}{%
4009 \glsadd[#1]{\glo@entry}%
4010 }%
4011 }
```

`\glsaddallunused`

`\glsaddallunused` [*glossary type*]

Add all used terms defined for the listed glossaries (without displaying any text). If optional argument is omitted, apply to all glossary types. This should typically go at the end of the document.

```
4012 \newrobustcmd*{\glsaddallunused}[1] [\glo@types] {%
4013 \forallglsentries[#1]{\glo@entry}%
4014 {%
4015 \ifglsused{\glo@entry}{\glsadd[format=@gobble]{\glo@entry}}%
4016 }%
4017 }
```

1.12 Creating associated files

The `\writeist` command creates the associated customized `.ist` `makeindex` style file. While defining this command, some characters have their catcodes temporarily changed to ensure they get written to the `.ist` file correctly. The `makeindex` actual character (usually `@`) is redefined to be a `?`, to allow internal commands to be written to the glossary file output file.

The special characters are stored in `\gls@actualchar`, `\gls@encapchar`, `\gls@levelchar` and `\gls@quotechar` to make them easier to use later, but don't change these values, because the characters are encoded in the command definitions that are used to escape the special characters (which means that the user no longer needs to worry about `makeindex` special characters).

The symbols and numbers label for group headings are hardwired into the .ist file as `glssymbols` and `glsnumbers`, the group titles can be translated (so that `glssymbolsgroupname` replaces `glssymbols` and `glsnumbersgroupname` replaces `glsnumbers`) using the command `glssetgrouptitle` which is defined in . This is done to prevent any problem characters in `glssymbolsgroupname` and `glsnumbersgroupname` from breaking hyperlinks.

`\glsopenbrace` Define `\glsopenbrace` to make it easier to write an opening brace to a file.

```
4018 \edef\glsopenbrace{\expandafter\@gobble\string\{}}
```

`\glsclosebrace` Define `\glsclosebrace` to make it easier to write an opening brace to a file.

```
4019 \edef\glsclosebrace{\expandafter\@gobble\string\}}
```

`\glsbackslash` Define `\glsbackslash` to make it easier to write a backslash to a file.

```
4020 \edef\glsbackslash{\expandafter\@gobble\string\}
```

`\glsquote` Define command that makes it easier to write quote marks to a file in the event that the double quote character has been made active.

```
4021 \edef\glsquote#1{\string"#1\string"}
```

`\glspercentchar` Define `\glspercentchar` to make it easier to write a percent character to a file.

```
4022 \edef\glspercentchar{\expandafter\@gobble\string\%}
```

`\glsildechar` Define `\glsildechar` to make it easier to write a tilde character to a file.

```
4023 \edef\glsildechar{\string~}
```

`\@glsfirstletter` Define the first letter to come after the digits 0,...,9. Only required for xindy.

```
4024 \ifglxindy
```

```
4025 \newcommand*{\@glsfirstletter}{A}
```

```
4026 \fi
```

`stLetterAfterDigits` Sets the first letter to come after the digits 0,...,9.

```
4027 \ifglxindy
```

```
4028 \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
```

```
4029 \renewcommand*{\@glsfirstletter}{#1}}
```

```
4030 \else
```

```
4031 \newcommand*{\GlsSetXdyFirstLetterAfterDigits}[1]{%
```

```
4032 \glsnoxindywarning\GlsSetXdyFirstLetterAfterDigits}
```

```
4033 \fi
```

`\@glsminrange` Define the minimum number of successive location references to merge into a range.

```
4034 \newcommand*{\@glsminrange}{2}
```

etXdyMinRangeLength Set the minimum range length. The value must either be none or a positive integer. The glossaries package doesn't check if the argument is valid, that is left to xindy.

```
4035 \ifglxindy
4036   \newcommand*\GlsSetXdyMinRangeLength}[1]{%
4037     \renewcommand*\@glsminrange}{#1}}
4038 \else
4039   \newcommand*\GlsSetXdyMinRangeLength}[1]{%
4040     \glsnoxindywarning\GlsSetXdyMinRangeLength}
4041 \fi
```

`\writeist`

```
4042 \ifglxindy
    Code to use if xindy is required.
4043   \def\writeist{%
    Define write register if not already defined
4044     \ifundef\glswrite{\newwrite\glswrite}{}%
    Update attributes list
4045     \@gls@addpredefinedattributes
    Open the file.
4046     \openout\glswrite=\istfilename
    Write header comment at the start of the file
4047     \write\glswrite{;; xindy style file created by the glossaries
4048       package}%
4049     \write\glswrite{;; for document '\jobname' on
4050       \the\year-\the\month-\the\day}%
    Specify the required styles
4051     \write\glswrite{^^J; required styles^^J}
4052     \@for\@xdystyle:=\@xdyrequiredstyles\do{%
4053       \ifx\@xdystyle\@empty
4054       \else
4055         \protected@write\glswrite}{(require
4056           \string"\@xdystyle.xdy\string")}%
4057       \fi
4058     }%
    List the allowed attributes (possible values used by the format key)
4059     \write\glswrite{^^J%
4060       ; list of allowed attributes (number formats)^^J}%
4061     \write\glswrite{(define-attributes ((\@xdyattributes)))}%
    Define any additional alphabets
4062     \write\glswrite{^^J; user defined alphabets^^J}%
4063     \write\glswrite{\@xdyuseralphabets}%
    Define location classes.
4064     \write\glswrite{^^J; location class definitions^^J}%
```


As from version 3.0, locations are now specified as $\langle Hprefix \rangle \langle number \rangle$, so need to add all possible combinations of location types.

```
4065 \@for\@gls@classI:=\@gls@xdy@locationlist\do{%
```

Case were $\langle Hprefix \rangle$ is empty:

```
4066 \protected@write\glswrite{}{(define-location-class
4067 \string"\@gls@classI\string"^^J\space\space\space
4068 (
4069 :sep "{ }{"
4070 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4071 :sep "}"
4072 )
4073 ^^J\space\space\space
4074 :min-range-length \@glsminrange^^J%
4075 )
4076 }%
```

Nested iteration over all classes:

```
4077 {%
4078 \@for\@gls@classII:=\@gls@xdy@locationlist\do{%
4079 \protected@write\glswrite{}{(define-location-class
4080 \string"\@gls@classII-\@gls@classI\string"
4081 ^^J\space\space\space
4082 (
4083 :sep "{"
4084 \csname @gls@xdy@Lclass@\@gls@classII\endcsname\space
4085 :sep " }{"
4086 \csname @gls@xdy@Lclass@\@gls@classI\endcsname\space
4087 :sep "}"
4088 )
4089 ^^J\space\space\space
4090 :min-range-length \@glsminrange^^J%
4091 )
4092 }%
4093 }%
4094 }%
4095 }%
```

User defined location classes (needs checking for new location format).

```
4096 \write\glswrite{^^J; user defined location classes}%
4097 \write\glswrite{\@xdyuserlocationdefs}%
```

Cross-reference class. (The unverified option is used as the cross-references are supplied using the list of labels along with the optional argument for $\backslash\text{glsseeformat}$ which xindy won't recognise.)

```
4098 \write\glswrite{^^J; define cross-reference class^^J}%
4099 \write\glswrite{(define-crossref-class \string"see\string"
4100 :unverified )}%
```

Define how cross-references should be displayed. This adds an empty set of braces after the cross-referencing information allowing for the final argument

of `\glsseeformat` which gets ignored. (When using `makeindex` this final argument contains the location information which is not required.)

```
4101 \write\glswrite{(markup-crossref-list
4102     :class \string"see\string"^^J\space\space\space
4103     :open \string"\string\glsseeformat\string"
4104     :close \string"{}\string")}%
```

List the order to sort the classes.

```
4105 \write\glswrite{^^J; define the order of the location classes}%
4106 \write\glswrite{(define-location-class-order
4107     (\@xdylocationclassorder))}%
```

Specify what to write to the start and end of the glossary file.

```
4108 \write\glswrite{^^J; define the glossary markup^^J}%
4109 \write\glswrite{(markup-index^^J\space\space\space
4110     :open \string"\string
4111     \glossarysection[\string\glossarytoctitle]{\string
4112     \glossarytitle}\string\glossarypreamble}%
```

Add all the xindy-only macro definitions (needed to prevent errors in the event that the user changes from xindy to `makeindex`)

```
4113 \@for\@this@ctr:=\@xdycounters\do{%
4114     {%
4115         \@for\@this@attr:=\@xdyattributelist\do{%
4116             \protected\write\glswrite{}{\string\providecommand*%
4117                 \expandafter\string
4118                 \csname glsX\@this@ctr X\@this@attr\endcsname[2]%
4119                 {%
4120                     \string\setentrycounter
4121                     [\expandafter\@gobble\string\#1]{\@this@ctr}%
4122                     \expandafter\string
4123                     \csname\@this@attr\endcsname
4124                     {\expandafter\@gobble\string\#2}%
4125                 }%
4126             }%
4127         }%
4128     }%
4129 }%
```

Add the end part of the open tag and the rest of the markup-index information:

```
4130 \write\glswrite{%
4131     \string\begin
4132     {theglossary}\string\glossaryheader\glstildechar n\string" ^^J\space
4133     \space\space:close \string"\glspercentchar\glstildechar n\string
4134     \end{theglossary}\string\glossarypostamble
4135     \glstildechar n\string" ^^J\space\space\space
4136     :tree}}%
```

Specify what to put between letter groups

```
4137 \write\glswrite{(markup-letter-group-list
4138     :sep \string"\string\glsgroupskip\glstildechar n\string")}%
```

Specify what to put between entries

```
4139 \write\glswrite{(markup-indexentry
4140 :open \string"\string\relax \string\glsresetentrylist
4141 \glstildechar n\string")}%
```

Specify how to format entries

```
4142 \write\glswrite{(markup-locclass-list :open
4143 \string"\glsopenbrace\string\glossaryentrynumbers
4144 \glsopenbrace\string\relax\space \string"^^J\space\space\space
4145 :sep \string", \string"
4146 :close \string"\glsclosebrace\glsclosebrace\string")}%
```

Specify how to separate location numbers

```
4147 \write\glswrite{(markup-locref-list
4148 :sep \string"\string\delimN\space\string")}%
```

Specify how to indicate location ranges

```
4149 \write\glswrite{(markup-range
4150 :sep \string"\string\delimR\space\string")}%
```

Specify 2-page and 3-page suffixes, if defined. First, the values must be sanitized to write them explicitly.

```
4151 \@onelevel@sanitize\gls@suffixF
4152 \@onelevel@sanitize\gls@suffixFF
4153 \ifx\gls@suffixF\@empty
4154 \else
4155 \write\glswrite{(markup-range
4156 :close "\gls@suffixF" :length 1 :ignore-end)}%
4157 \fi
4158 \ifx\gls@suffixFF\@empty
4159 \else
4160 \write\glswrite{(markup-range
4161 :close "\gls@suffixFF" :length 2 :ignore-end)}%
4162 \fi
```

Specify how to format locations.

```
4163 \write\glswrite{^^J; define format to use for locations^^J}%
4164 \write\glswrite{\@xdylocref}%
```

Specify how to separate letter groups.

```
4165 \write\glswrite{^^J; define letter group list format^^J}%
4166 \write\glswrite{(markup-letter-group-list
4167 :sep \string"\string\glsgroupskip\glstildechar n\string")}%
```

Define letter group headings.

```
4168 \write\glswrite{^^J; letter group headings^^J}%
4169 \write\glswrite{(markup-letter-group
4170 :open-head \string"\string\glsgroupheading
4171 \glsopenbrace\string"^^J\space\space\space
4172 :close-head \string"\glsclosebrace\string")}%
```

Define additional letter groups.

```
4173 \write\glswrite{^^J; additional letter groups^^J}%  
4174 \write\glswrite{\@xdylettergroups}%
```

Define additional sort rules

```
4175 \write\glswrite{^^J; additional sort rules^^J}  
4176 \write\glswrite{\@xdysortrules}%
```

Close the style file

```
4177 \closeout\glswrite
```

Suppress any further calls.

```
4178 \let\writeist\relax  
4179 }  
4180 \else
```

Code to use if makeindex is required.

```
4181 \edef\@gls@actualchar{\string?}  
4182 \edef\@gls@encapchar{\string|}  
4183 \edef\@gls@levelchar{\string!}  
4184 \edef\@gls@quotechar{\string"}  
4185 \def\writeist{\relax  
4186 \ifundef\glswrite{\newwrite\glswrite}{\relax  
4187 \openout\glswrite=\istfilename  
4188 \write\glswrite{\glspercentchar\space makeindex style file  
4189 created by the glossaries package}  
4190 \write\glswrite{\glspercentchar\space for document  
4191 '\jobname' on \the\year-\the\month-\the\day}  
4192 \write\glswrite{actual '\@gls@actualchar'}  
4193 \write\glswrite{encap '\@gls@encapchar'}  
4194 \write\glswrite{level '\@gls@levelchar'}  
4195 \write\glswrite{quote '\@gls@quotechar'}  
4196 \write\glswrite{keyword \string"\string\glossaryentry\string"}  
4197 \write\glswrite{preamble \string"\string\glossarysection[\string  
4198 \glossarytoctitle]{\string\glossarytitle}\string  
4199 \glossarypreamble\string\n\string\begin{theglossary}\string  
4200 \glossaryheader\string\n\string"}  
4201 \write\glswrite{postamble \string"\string%\string\n\string  
4202 \end{theglossary}\string\glossarypostamble\string\n  
4203 \string"}  
4204 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n  
4205 \string"}  
4206 \write\glswrite{item_0 \string"\string%\string\n\string"}  
4207 \write\glswrite{item_1 \string"\string%\string\n\string"}  
4208 \write\glswrite{item_2 \string"\string%\string\n\string"}  
4209 \write\glswrite{item_01 \string"\string%\string\n\string"}  
4210 \write\glswrite{item_x1  
4211 \string"\string\relax \string\glsresetentrylist\string\n  
4212 \string"}  
4213 \write\glswrite{item_12 \string"\string%\string\n\string"}  
4214 \write\glswrite{item_x2
```

```

4215     \string"\string\relax \string\glsresetentrylist\string\n
4216     \string"}

4217     \write\glswrite{delim_0 \string"\string\{\string
4218         \glossaryentrynumbers\string\{\string\relax \string"}
4219     \write\glswrite{delim_1 \string"\string\{\string
4220         \glossaryentrynumbers\string\{\string\relax \string"}
4221     \write\glswrite{delim_2 \string"\string\{\string
4222         \glossaryentrynumbers\string\{\string\relax \string"}
4223     \write\glswrite{delim_t \string"\string\}\string\}\string"}
4224     \write\glswrite{delim_n \string"\string\delimN \string"}
4225     \write\glswrite{delim_r \string"\string\delimR \string"}
4226     \write\glswrite{headings_flag 1}
4227     \write\glswrite{heading_prefix
4228         \string"\string\glsgroupheading\string\{\string"}
4229     \write\glswrite{heading_suffix
4230         \string"\string\}\string\relax
4231         \string\glsresetentrylist \string"}
4232     \write\glswrite{symhead_positive \string"glssymbols\string"}
4233     \write\glswrite{numhead_positive \string"glssymbols\string"}
4234     \write\glswrite{page_compositor \string"glscpositor\string"}
4235     \@gls@escbsdq\gls@suffixF
4236     \@gls@escbsdq\gls@suffixFF
4237     \ifx\gls@suffixF\@empty
4238     \else
4239         \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
4240     \fi
4241     \ifx\gls@suffixFF\@empty
4242     \else
4243         \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
4244     \fi
4245     \closeout\glswrite
4246     \let\writeist\relax
4247 }
4248 \fi

```

The command `\noist` will suppress the creation of the `.ist` file. Obviously you need to use this command before `\writeist` to have any effect.

`\noist`

```

4249 \newcommand{\noist}{%
    Update attributes list
4250     \@gls@addpredefinedattributes
4251     \let\writeist\relax
4252 }

```

`\@makeglossary` is an internal command that takes an argument indicating the glossary type. This command will create the glossary file required by `makeindex` for the given glossary type, using the extension supplied by

the `<out-ext>` parameter used in `\newglossary` (and it will also activate the `\glossary` command, and create the customized `.ist` makeindex style file).

Note that you can't use `\@makeglossary` for only some of the defined glossaries. You either need to have a `\makeglossary` for all glossaries or none (otherwise you will end up with a situation where \TeX is trying to write to a non-existent file). The relevant glossary must be defined prior to using `\@makeglossary`.

`\@makeglossary`

```

4253 \newcommand*\@makeglossary}[1]{%
4254   \ifglossaryexists{#1}%
4255   {%

      Only create a new write if savewrites=false otherwise create a token to collect
      the information.

4256     \ifglssavewrites
4257       \expandafter\newtoks\csname glo@#1@filetok\endcsname
4258     \else
4259       \expandafter\newwrite\csname glo@#1@file\endcsname
4260       \expandafter\glsopenfile\csname glo@#1@file\endcsname{#1}%
4261     \fi
4262     \@gls@renewglossary
4263     \writeist
4264   }%
4265   {%
4266     \PackageError{glossaries}%
4267     {Glossary type ‘#1’ not defined}%
4268     {New glossaries must be defined before using \string\makeglossary}%
4269   }%
4270 }

```

`\@glsopenfile` Open write file associated with the given glossary.

```

4271 \newcommand*\@glsopenfile}[2]{%
4272   \immediate\openout#1=\jobname.\csname @glotype@#2@out\endcsname
4273   \PackageInfo{glossaries}{Writing glossary file
4274     \jobname.\csname @glotype@#2@out\endcsname}%
4275 }

```

`\@closegls`

```

4276 \newcommand*\@closegls}[1]{%
4277   \closeout\csname glo@#1@file\endcsname
4278 }
4279 % \end{macrocode}
4280 %\end{macro}
4281 %
4282 %\begin{macro}{\@gls@automake}
4283 %\changes{4.08}{2014-07-30}{new}
4284 % \begin{macrocode}
4285 \ifglsxindy

```

```

4286 \newcommand*{\@gls@automake}[1]{%
4287   \ifglossaryexists{#1}
4288   {%
4289     \@closegls{#1}%
4290     \ifdefstring{\glsorder}{letter}%
4291     {\def\@gls@order{-M ord/letorder }}%
4292     {\let\@gls@order\@empty}%
4293     \ifcsundef{@xdy@#1@language}%
4294     {\let\@gls@langmod\@xdy@main@language}%
4295     {\letcs\@gls@langmod{@xdy@#1@language}}%
4296     \edef\@gls@dothiswrite{\noexpand\write18{xindy
4297       -I xindy
4298       \@gls@order
4299       -L \@gls@langmod\space
4300       -M \gls@istfilebase\space
4301       -C \gls@codepage\space
4302       -t \jobname.\csuse{@glotype@#1@log}
4303       -o \jobname.\csuse{@glotype@#1@in}
4304       \jobname.\csuse{@glotype@#1@out}}}%
4305     }%
4306     \@gls@dothiswrite
4307   }%
4308   {%
4309     \GlossariesWarning{Can't make glossary '#1', it doesn't exist}%
4310   }%
4311 }
4312 \else
4313 \newcommand*{\@gls@automake}[1]{%
4314   \ifglossaryexists{#1}
4315   {%
4316     \@closegls{#1}%
4317     \ifdefstring{\glsorder}{letter}%
4318     {\def\@gls@order{-l }}%
4319     {\let\@gls@order\@empty}%
4320     \edef\@gls@dothiswrite{\noexpand\write18{makeindex \@gls@order
4321       -s \istfilename\space
4322       -t \jobname.\csuse{@glotype@#1@log}
4323       -o \jobname.\csuse{@glotype@#1@in}
4324       \jobname.\csuse{@glotype@#1@out}}}%
4325     }%
4326     \@gls@dothiswrite
4327   }%
4328   {%
4329     \GlossariesWarning{Can't make glossary '#1', it doesn't exist}%
4330   }%
4331 }
4332 \fi

```

rn@nomakeglossaries Issue warning that \makeglossaries hasn't been used.

```
4333 \newcommand*{\@warn@nomakeglossaries}{}
```

Only use this if warning if `\printglossary` has been used without `\makeglossaries`

```
4334 \newcommand*{\@warn@nomakeglossaries}{\@warn@nomakeglossaries}
```

`\makeglossaries` will use `\@makeglossary` for each glossary type that has been defined. New glossaries need to be defined before using `\makeglossary`, so have `\makeglossaries` redefine `\newglossary` to prevent it being used afterwards.

`\makeglossaries`

```
4335 \newcommand*{\makeglossaries}{%
```

Define the write used for style file also used for all other output files if `savewrites=true`.

```
4336 \ifundef{glswrite}{\newwrite\glswrite}{}%
```

If the user removes the glossary package from their document, ensure the next run doesn't throw a load of undefined control sequence errors when the aux file is parsed.

```
4337 \protected@write\@auxout{}{\string\providecommand\string\@glsorder[1]{}}
```

```
4338 \protected@write\@auxout{}{\string\providecommand\string\@istfilename[1]{}}
```

Write the name of the style file to the aux file (needed by `makeglossaries`)

```
4339 \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
```

```
4340 \protected@write\@auxout{}{\string\@glsorder{\glsorder}}%
```

Iterate through each glossary type and activate it.

```
4341 \@for\@glo@type:=\@glo@types\do{%
```

```
4342 \ifthenelse{\equal{\@glo@type}{} }{}{}%
```

```
4343 \@makeglossary{\@glo@type}}%
```

```
4344 }%
```

New glossaries must be created before `\makeglossaries` so disable `\newglossary`.

```
4345 \renewcommand*\newglossary[4] [] {}%
```

```
4346 \PackageError{glossaries}{New glossaries
```

```
4347 must be created before \string\makeglossaries}{You need
```

```
4348 to move \string\makeglossaries\space after all your
```

```
4349 \string\newglossary\space commands}}%
```

Any subsequent instances of this command should have no effect

```
4350 \let\@makeglossary\relax
```

```
4351 \let\makeglossary\relax
```

```
4352 \let\makeglossaries\relax
```

Disable all commands that have no effect after `\makeglossaries`

```
4353 \@disable@onlypremakeg
```

Allow see key:

```
4354 \let\gls@checkseeallowed\relax
```

Suppress warning about no `\makeglossaries`

```
4355 \let\warn@nomakeglossaries\relax
```


Activate warning about missing `\printglossary`

```

4356 \def\warn@noprintglossary{%
4357   \GlossariesWarningNoLine{No \string\printglossary\space
4358     or \string\printglossaries\space
4359     found.^^J(Remove \string\makeglossaries\space if you don't want
4360     any glossaries.)^^JThis document will not have a glossary}%
4361 }%

Declare list parser for \glsdisplaynumberlist
4362 \ifglssavenumberlist
4363   \edef\@gls@dodolistparser{\noexpand\DeclareListParser
4364     {\noexpand\glsnumlistparser}{\delimN}}%
4365   \@gls@dodolistparser
4366 \fi

Prevent user from also using \makenoidxglossaries
4367 \let\makenoidxglossaries\@no@makeglossaries

Prohibit sort key in printgloss family:
4368 \renewcommand*{\@printgloss@setsort}{%
4369   \let\@glo@assign@sortkey\@glo@no@assign@sortkey
4370 }%

Check the automake setting:
4371 \ifglsautomake
4372   \renewcommand*{\@gls@doautomake}{%
4373     \@for\@gls@type:=\@glo@types\do{%
4374       \ifdefempty{\@gls@type}{}%
4375       {\@gls@automake{\@gls@type}}%
4376     }%
4377   }%
4378 \fi
4379 }

Must occur in the preamble:
4380 \@onlypreamble{\makeglossaries}

```

`\glswrite` The definition of `\glswrite` has now been moved to `\makeglossaries` so that it's only defined if needed.

The `\makeglossary` command is redefined to be identical to `\makeglossaries`. (This is done to reinforce the message that you must either use `\@makeglossary` for all the glossaries or for none of them.)

`\makeglossary`

```

4381 \let\makeglossary\makeglossaries

```

If `\makeglossaries` hasn't been used, issue a warning. Also issue a warning if neither `\printglossaries` nor `\printglossary` have been used.

```

4382 \AtEndDocument{%
4383   \warn@nomakeglossaries

```

```
4384 \warn@noprintglossary
4385 }
```

makenoidxglossaries Analogous to \makeglossaries this activates the commands needed for \printnoidxglossary

```
4386 \newcommand*\makenoidxglossaries{%
```

Redefine empty glossary warning:

```
4387 \renewcommand{\@gls@nofwarn}[1]{%
4388 \GlossariesWarning{Empty glossary for
4389 \string\printnoidxglossary[type={##1}].
4390 Rerun may be required (or you may have forgotten to use
4391 commands like \string\gls).}%
4392 }%
```

Don't escape makeindex/xindy characters

```
4393 \let\@gls@checkmkidxchars\@gobble
```

Write glossary information to aux instead of glossary files

```
4394 \let\@do@ewrglossary\gls@noidxglossary
```

Switch on group headings that use the character code:

```
4395 \let\@gls@getgrouptitle\@gls@noidx@getgrouptitle
```

Allow see key:

```
4396 \let\gls@checkseeallowed\relax
```

Redefine cross-referencing macro:

```
4397 \renewcommand{\@do@seeglossary}[2]{%
4398 \edef\@gls@label{\glsdetoklabel{##1}}%
4399 \protected@write\@auxout{}{%
4400 \string\@gls@reference
4401 {\csname glo@\@gls@label @type\endcsname}%
4402 {\@gls@label}%
4403 {%
4404 \string\glsseeformat##2}%
4405 }%
4406 }%
4407 }%
```

If user removes the glossaries package from their document, ensure the next run doesn't throw a load of undefined control sequence errors when the aux file is parsed.

```
4408 \AtBeginDocument
4409 {%
4410 \write\@auxout{\string\providecommand\string\@gls@reference[3]{}%
4411 }%
```

Change warning about no glossares

```
4412 \def\warn@noprintglossary{%
4413 \GlossariesWarningNoLine{No \string\printnoidxglossary\space
4414 or \string\printnoidxglossaries ^^J
4415 found. (Remove \string\makenoidxglossaries\space if you
```

```

4416     don't want any glossaries.)^^JThis document will not have a glossary}%
4417 }%

  Suppress warning about no \makeglossaries
4418 \let\warn@nomakeglossaries\relax

  Prevent user from also using \makeglossaries
4419 \let\makeglossaries\@no@makeglossaries

  Allow sort key in printgloss family:
4420 \renewcommand*{\@printgloss@setsort}{%
4421 \let\@glo@assign@sortkey\@glo@assign@sortkey

  Initialise default sort order:
4422 \def\@glo@sorttype{\@glo@default@sorttype}%
4423 }%

  All entries must be defined in the preamble:
4424 \renewcommand*\new@glossaryentry[2]{%
4425 \PackageError{glossaries}{Glossary entries must be
4426 defined in the preamble^^Jwhen you use
4427 \string\makenoidxglossaries}%
4428 {Either move your definitions to the preamble or use
4429 \string\makeglossaries}%
4430 }%

  Redefine \glsentrynumberlist
4431 \renewcommand*\glsentrynumberlist[1]{%
4432 \letcs{\@gls@loclist}{glo@\glsdetoklabel{##1}@loclist}%
4433 \ifdef\@gls@loclist
4434 {%
4435 \glsnoidxloclist{\@gls@loclist}%
4436 }%
4437 {%
4438 \ifglsentryexists{##1}%
4439 {%
4440 \GlossariesWarning{Missing location list for '##1'. Either
4441 a rerun is required or you haven't referenced the entry.}%
4442 }%
4443 {%
4444 \PackageError{glossaries}{Glossary entry '##1' has not been
4445 defined.}{}%
4446 }%
4447 }%
4448 }%

  Redefine \glsdisplaynumberlist
4449 \renewcommand*\glsdisplaynumberlist[1]{%
4450 \letcs{\@gls@loclist}{glo@\glsdetoklabel{##1}@loclist}%
4451 \ifdef\@gls@loclist
4452 {%
4453 \def\@gls@noidxloclist@sep{%

```

```

4454     \def\@gls@noidxloclist@sep{%
4455         \def\@gls@noidxloclist@sep{%
4456             \glsnumlistsep
4457         }%
4458     \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
4459     }%
4460 }%
4461 \def\@gls@noidxloclist@finalsep{}%
4462 \def\@gls@noidxloclist@prev{}%
4463 \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
4464 \@gls@noidxloclist@finalsep
4465 \@gls@noidxloclist@prev
4466 }%
4467 {%
4468     ??\ifglsentryexists{##1}%
4469     {%
4470         \GlossariesWarning{Missing location list for ‘##1’. Either
4471             a rerun is required or you haven’t referenced the entry.}%
4472     }%
4473     {%
4474         \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4475             defined.}{}%
4476     }%
4477 }%
4478 }%

```

Provide a generic way of iterating through the number list:

```

4479 \renewcommand*\glsnumberlistloop}[3]{%
4480     \letcs{\@gls@loclist}{glo@\glsdetoklabel{##1}@loclist}%
4481     \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
4482     \let\@gls@org@glsseeformat\glsseeformat
4483     \let\glsnoidxdisplayloc##2\relax
4484     \let\glsseeformat##3\relax
4485     \ifdef\@gls@loclist
4486     {%
4487         \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
4488     }%
4489     {%
4490         \ifglsentryexists{##1}%
4491         {%
4492             \GlossariesWarning{Missing location list for ‘##1’. Either
4493                 a rerun is required or you haven’t referenced the entry.}%
4494         }%
4495         {%
4496             \PackageError{glossaries}{Glossary entry ‘##1’ has not been
4497                 defined.}{}%
4498         }%
4499     }%
4500     \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
4501     \let\glsseeformat\@gls@org@glsseeformat

```

```

4502 }%
      Modify sanitize sort function
4503 \let\@gls@sanitizesort\@gls@noidx@sanitizesort
4504 \let\@gls@nosanitizesort\@gls@noidx@nosanitizesort
4505 \@gls@noidx@setsanitizesort
4506 }
      Preamble-only command:
4507 \@onlypreamble{\makenoidxglossaries}

```

```

\glsnumberlistloop \glsnumberlistloop{<label>}{<handler>}

```

```

4508 \newcommand*\glsnumberlistloop}[2]{%
4509   \PackageError{glossaries}{\string\glsnumberlistloop\space
4510     only works with \string\makenoidxglossaries}{}%
4511 }

```

```

numberlistloophandler Handler macro for \glsnumberlistloop. (The argument should be in the
      form \glsnoidxdisplayloc{<prefix>}{<counter>}{<format>}{<n>})
4512 \newcommand*\glsnoidxnumberlistloophandler}[1]{%
4513   #1%
4514 }

```

```

\@no@makeglossaries Can't use both \makeglossaries and \makenoidxglossaries
4515 \newcommand*\@no@makeglossaries}{%
4516   \PackageError{glossaries}{You can't use both
4517     \string\makeglossaries\space and \string\makenoidxglossaries}%
4518   {Either use one or other (or none) of those commands but not both
4519     together.}%
4520 }

```

```

\@gls@noref@warn Warning when no instances of \@gls@reference found.
4521 \newcommand{\@gls@noref@warn}[1]{%
4522   \GlossariesWarning{\string\makenoidxglossaries\space
4523     is required to make \string\printnoidxglossary[type={#1}] work}%
4524 }

```

```

\gls@noidxglossary Write the glossary information to the aux file:
4525 \newcommand*\gls@noidxglossary}{%
4526   \protected@write\@auxout{}{%
4527     \string\@gls@reference
4528     {\csname glo@\@gls@label @type\endcsname}%
4529     {\@gls@label}%
4530     {\string\glsnoidxdisplayloc
4531       {\@glo@counterprefix}%
4532       {\@gls@counter}%
4533       {\@glsnumberformat}%

```

```

4534         {\@glslocref}%
4535     }%
4536 }%
4537 }

```

1.13 Writing information to associated files

`\istfile` Deprecated.

```
4538 \def\istfile{\glswrite}
```

At the end of the document, the files should be created if `savewrites=true`.

```

4539 \AtEndDocument{%
4540     \glswritefiles
4541 }

```

`\@glswritefiles` Only write the files if `savewrites=true`

```
4542 \newcommand*{\@glswritefiles}{%
```

Iterate through all the glossaries

```
4543 \forallglossaries{\@glo@type}{%
```

Check for empty glossaries (patch provided by Patrick Häcker)

```

4544     \ifcsundef{glo@\@glo@type @filetok}%
4545     {%
4546         \def\gls@tmp{}%
4547     }%
4548     {%
4549         \edef\gls@tmp{\expandafter\the
4550             \csname glo@\@glo@type @filetok\endcsname}%
4551     }%
4552     \ifx\gls@tmp\@empty
4553         \ifx\@glo@type\glsdefaulttype
4554             \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
4555                 entries.^^JRemember to use package option ‘nomain’ if
4556 you
4557                 don’t want to^^Juse the main glossary}%
4558         \else
4559             \GlossariesWarningNoLine{Glossary ‘\@glo@type’ has no
4560                 entries}%
4561         \fi
4562     \else
4563         \@glsopenfile{\glswrite}{\@glo@type}%
4564         \immediate\write\glswrite{%
4565             \expandafter\the
4566                 \csname glo@\@glo@type @filetok\endcsname}%
4567         \immediate\closeout\glswrite
4568     \fi
4569 }%
4570 }

```

As from v4.10, the `\glossary` command is used by the `glossaries` package. Since the user isn't expected to use this command (as `glossaries` takes care of the particular format required for `makeindex/xindy`) there's no need for a user level command. Using a custom internal command prevents any conflict with other packages (and with the `\mark` mechanism).

In v4.10, the redefinition of `\glossary` was removed since it wasn't intended as a user level command, however it seems there are packages that have hacked the internal macros used by `glossaries` and no longer work with this redefinition removed, so it's been restored in v4.11 but is not used at all by `glossaries`. (This may be removed or moved to a compatibility mode in future.)

`\glossary`

```
4571 \if@gls@docloaded
4572 \else
4573   \renewcommand*{\glossary}[1][main]{\gls@glossary{#1}}
4574 \fi
```

The associated number should be stored in `\theglentrycounter` before using `\gls@glossary`.

`\gls@glossary`

```
4575 \newcommand*{\gls@glossary}[1]{%
4576   \@gls@glossary{#1}%
4577 }
```

`\@gls@glossary` (In v4.10, `\glossary` was redefined to `\@gls@glossary` to avoid conflict with other packages.) Define internal `\@gls@glossary` to ignore its argument. This gets redefined in `\@makeglossary`. This is defined to just `\index` as memoir changes the definition of `\@index`. (Thanks to Dan Luecking for pointing this out.) The argument #1 is the glossary type.

```
4578 \newcommand*{\@gls@glossary}[1]{\index}
```

This is a convenience command to set `\@gls@glossary`. It's used by `\@makeglossary` and then redefined to do nothing, as it only needs to be done once.

`\@gls@renewglossary`

```
4579 \newcommand{\@gls@renewglossary}{%
4580   \gdef\@gls@glossary##1{\@bsphack\beginngroup\gls@wrglossary{##1}}%
4581   \let\@gls@renewglossary\@empty
4582 }
```

The `\gls@wrglossary` command is defined to have two arguments. The first argument is the glossary type, the second argument is the glossary entry (the format of which is set in `\glslink`).

`\gls@wrglossary`

```
4583 \newcommand*{\gls@wrglossary}[2]{%
4584   \ifglssavewrites
```

```

4585 \protected@edef\@gls@tmp{\the\csname glo@#1@filetok\endcsname#2}%
4586 \expandafter\global\expandafter\csname glo@#1@filetok\endcsname
4587 \expandafter{\@gls@tmp^^J}%
4588 \else

4589 \ifcsdef{glo@#1@file}%
4590 {%
4591 \expandafter\protected@write\csname glo@#1@file\endcsname{%
4592 \gls@disablepagerefexpansion}{#2}%
4593 }%
4594 {%
4595 \ifignoredglossary{#1}{}%
4596 {%
4597 \GlossariesWarning{No file defined for glossary ‘#1’}%
4598 }%
4599 }%
4600 \fi
4601 \endgroup\@esphack
4602 }

```

\@do@wrglossary

```

4603 \newcommand*\@do@wrglossary}[1]{%
4604 \ifglsindexonlyfirst
4605 \ifglsused{#1}{\@do@wrglossary{#1}}%
4606 \else
4607 \@do@wrglossary{#1}%
4608 \fi
4609 }

```

\protected@pagefmts List of page formats to be protected against expansion.

```

4610 \newcommand*\gls@protected@pagefmts{%
4611 \gls@numberpage,\gls@alphpage,\gls@Alphpage,\gls@romanpage,\gls@Romanpage%
4612 }

```

\gls@disablepagerefexpansion

```

4613 \newcommand*\gls@disablepagerefexpansion{%
4614 \@for\@gls@this:=\gls@protected@pagefmts\do
4615 {%
4616 \expandafter\let\@gls@this\relax
4617 }%
4618 }

```

\gls@alphpage

```

4619 \newcommand*\gls@alphpage{\@alph\c@page}

```

\gls@Alphpage

```

4620 \newcommand*\gls@Alphpage{\@Alph\c@page}

```


`\gls@numberpage`

```
4621 \newcommand*{\gls@numberpage}{\number\c@page}
```

`\gls@romanpage`

```
4622 \newcommand*{\gls@romanpage}{\romannumeral\c@page}
```

`\gls@Romanpage`

```
4623 \newcommand*{\gls@Romanpage}{\@Roman\c@page}
```

`\glsaddprotectedpagefmt`

```
\glsaddprotectedpagefmt{<cs name>}
```

Added a page format to the list of protected page formats. The argument should be the name (without a backslash) of the command that takes a TeX register as the argument (`\<csname>\c@page` must be valid).

```
4624 \newcommand*{\glsaddprotectedpagefmt}[1]{%
4625   \eappto\gls@protected@pagefmts{\expandonce{\csname gls#1page\endcsname}}}%
4626   \csedef{gls#1page}{\expandonce{\csname#1\endcsname}\noexpand\c@page}%
4627   \eappto\@wrglossarynumberhook{%
4628     \noexpand\let\expandonce{\csname org@gls#1\endcsname}%
4629     \expandonce{\csname#1\endcsname}%
4630     \noexpand\def\expandonce{\csname#1\endcsname}{%
4631       \noexpand\@wrglossary@pageformat
4632       \expandonce{\csname gls#1page\endcsname}%
4633       \expandonce{\csname org@gls#1\endcsname}%
4634     }%
4635   }%
4636 }
```

`\@do@wrglossarynumberhook`

Hook used by `\@do@wrglossary`

```
4637 \newcommand*\@wrglossarynumberhook{}
```

`\@do@wrglossary@pageformat`

```
4638 \newcommand{\@wrglossary@pageformat}[3]{%
4639   \ifx#3\c@page #1\else #2#3\fi
4640 }
```

`\@do@wrglossary`

Write the glossary entry in the appropriate format. (Need to set `\@glsnumberformat` and `\@gls@counter` prior to use.) The argument is the entry's label.

```
4641 \newcommand*{\@do@wrglossary}[1]{%
4642   \begingroup
```

First a bit of hackery to prevent premature expansion of `\c@page`. Store original definitions:

```
4643   \let\orgthe\the
4644   \let\orgnumber\number
4645   \let\orgromannumeral\romannumeral
```

```

4646 \let\orgalph\@alph
4647 \let\orgAlph\@Alph
4648 \let\orgRoman\@Roman

```

Redefine:

```

4649 \def\the##1{%
4650   \ifx##1\c@page \gls@numberpage\else\orgthe##1\fi}%
4651 \def\number##1{%
4652   \ifx##1\c@page \gls@numberpage\else\orgnumber##1\fi}%
4653 \def\romannumeral##1{%
4654   \ifx##1\c@page \gls@romanpage\else\orgromannumeral##1\fi}%
4655 \def\@Roman##1{%
4656   \ifx##1\c@page \gls@Romanpage\else\orgRoman##1\fi}%
4657 \def\@alph##1{%
4658   \ifx##1\c@page \gls@alphpage\else\orgalph##1\fi}%
4659 \def\@Alph##1{%
4660   \ifx##1\c@page \gls@Alphpage\else\orgAlph##1\fi}%

```

Add hook to allow for other number formats:

```

4661 \@wrglossarynumberhook

```

Prevent expansion:

```

4662 \gls@disablepagerefexpansion

```

Now store location in \glslocref:

```

4663 \protected@xdef\@glslocref{\theglsentrycounter}%
4664 \endgroup

```

Escape any special characters

```

4665 \@gls@checkmkidxchars\@glslocref

```

Check if the hyper-location is the same as the location and set the hyper prefix.

```

4666 \expandafter\ifx\theHglSentrycounter\theglsentrycounter\relax
4667 \def\@glo@counterprefix{%
4668 \else
4669   \protected@edef\@glsHlocref{\theHglSentrycounter}%
4670   \@gls@checkmkidxchars\@glsHlocref
4671   \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
4672     {\@glslocref}{\@glsHlocref}}%
4673   }%
4674   \@do@gls@getcounterprefix
4675 \fi

```

De-tok label if required

```

4676 \edef\@gls@label{\glsdetoklabel{#1}}%

```

Write the information to file:

```

4677 \@do@@@wrglossary
4678 }

```

```

\@do@@@wrglossary

```

```

4679 \newcommand*{\@do@@@wrglossary}{%

```

Determine whether to use xindy or makeindex syntax

```
4680 \ifglxindy
```

Need to determine if the formatting information starts with a (or) indicating a range.

```
4681 \expandafter\@glo@check@mkidxrangear\@glsnumberformat\@nil
4682 \def\@glo@range{}%
4683 \expandafter\if\@glo@prefix(\relax
4684 \def\@glo@range{:open-range}%
4685 \else
4686 \expandafter\if\@glo@prefix)\relax
4687 \def\@glo@range{:close-range}%
4688 \fi
4689 \fi
```

Write to the glossary file using xindy syntax.

```
4690 \gls@glossary{\csname glo@\@gls@label @type\endcsname}{%
4691 (indexentry :tkey (\csname glo@\@gls@label @index\endcsname)
4692 :locref \string"{\@glo@counterprefix}{\@glslocref}\string" %
4693 :attr \string"\@gls@counter\@glo@suffix\string"
4694 \@glo@range
4695 )
4696 }%
4697 \else
```

Convert the format information into the format required for makeindex

```
4698 \@set@glo@numformat{\@glo@numfmt}{\@gls@counter}{\@glsnumberformat}%
4699 {\@glo@counterprefix}%
```

Write to the glossary file using makeindex syntax.

```
4700 \gls@glossary{\csname glo@\@gls@label @type\endcsname}{%
4701 \string\glossaryentry{\csname glo@\@gls@label @index\endcsname
4702 \@gls@encapchar\@glo@numfmt}{\@glslocref}}%
4703 \fi
4704 }
```

`ls@getcounterprefix` Get the prefix that needs to be prepended to counter in order to get the hyper counter. (For example, with the standard article class and hyperref, `\theequation` needs to be prefixed with `<section num>|.` to get the equivalent `\theHequation`.) NB this assumes that the prefix ends with a dot, which is the standard. (Otherwise it makes the xindy location classes more complicated.)

```
4705 \newcommand*\@gls@getcounterprefix[2]{%
4706 \edef\@gls@thisloc{#1}\edef\@gls@thisHloc{#2}%
4707 \ifx\@gls@thisloc\@gls@thisHloc
4708 \def\@glo@counterprefix{}%
4709 \else
4710 \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
4711 \def\@glo@tmp{##2}%
4712 \ifx\@glo@tmp\@empty
```

```

4713     \def\@glo@counterprefix{}%
4714     \else
4715     \def\@glo@counterprefix{##1}%
4716     \fi
4717   }%
4718   \@gls@get@counterprefix#2.#1\end@getprefix

```

Warn if no prefix can be formed.

```

4719   \ifx\@glo@counterprefix\@empty
4720     \GlossariesWarning{Hyper target ‘#2’ can’t be formed by
4721     prefixing location ‘#1’. You need to modify the
4722     definition of \string\theH\@gls@counter otherwise you
4723     will get the warning: “name{\@gls@counter.#1}’ has been
4724     referenced but does not exist”}%
4725   \fi
4726   \fi
4727 }

```

1.14 Glossary Entry Cross-References

`\do@seeglossary` Write the glossary entry with a cross reference. The first argument is the entry’s label, the second must be in the form `[\langle tag \rangle]{\langle list \rangle}`, where `\langle tag \rangle` is a tag such as “see” and `\langle list \rangle` is a list of labels.

```

4728 \newcommand{\do@seeglossary}[2]{%
4729 \def\@gls@xref{#2}%
4730 \@onelevel@sanitize\@gls@xref
4731 \@gls@checkmkidxchars\@gls@xref
4732 \ifglsxindy
4733   \gls@glossary{\csname glo@#1@type\endcsname}{%
4734     (indexentry
4735       :tkey (\csname glo@#1@index\endcsname)
4736       :xref (\string"\@gls@xref\string")
4737       :attr \string"see\string"
4738     )
4739   }%
4740 \else
4741   \gls@glossary{\csname glo@#1@type\endcsname}{%
4742     \string\glossaryentry{\csname glo@#1@index\endcsname
4743     \@gls@encapchar glsseeformat\@gls@xref}{Z}}%
4744 \fi
4745 }

```

`\@gls@fixbraces` If no optional argument is specified, list needs to be enclosed in a set of braces.

```

4746 \def\@gls@fixbraces#1#2#3\@nil{%
4747   \ifx#2[\relax
4748     \@gls@fixbraces#1#2#3\@end@fixbraces
4749   \else
4750     \def#1{{#2#3}}%
4751   \fi

```

4752 }

`\@gls@fixbraces`

```
4753 \def\@gls@fixbraces#1[#2]#3\@end@fixbraces{%  
4754   \def#1{[#2]{#3}}%  
4755 }
```

`\glssee` `\glssee{<label>}{<cross-ref list>}`

```
4756 \DeclareRobustCommand*\glssee}[3][\seename]{%  
4757   \@do@seeglossary{#2}{#1}{#3}}%  
4758 \newcommand*\@glssee}[3][\seename]{%  
4759   \glssee[#1]{#3}{#2}}
```

`\glsseeformat` The first argument specifies what tag to use (e.g. “see”), the second argument is a comma-separated list of labels. The final argument (the location) is ignored.

```
4760 \DeclareRobustCommand*\glsseeformat}[3][\seename]{%  
4761   \emph{#1} \glsseelist{#2}}
```

`\glsseelist` `\glsseelist{<list>}` formats list of entry labels.

```
4762 \DeclareRobustCommand*\glsseelist}[1]{%
```

If there is only one item in the list, set the last separator to do nothing.

```
4763   \let\@gls@dolast\relax
```

Don't display separator on the first iteration of the loop

```
4764   \let\@gls@donext\relax
```

Iterate through the labels

```
4765   \@for\@gls@thislabel:=#1\do{%
```

Check if on last iteration of loop

```
4766     \ifx\@xfor@nextelement\@nnil
```

```
4767       \@gls@dolast
```

```
4768     \else
```

```
4769       \@gls@donext
```

```
4770     \fi
```

Display the entry for this label. (Expanding label as it's a temporary control sequence that's used elsewhere.)

```
4771     \expandafter\glsseeitem\expandafter{\@gls@thislabel}%
```

Update separators

```
4772     \let\@gls@dolast\glsseelastsep
```

```
4773     \let\@gls@donext\glsseesep
```

```
4774   }%
```

```
4775 }
```

`\glsseelastsep` Separator to use between penultimate and ultimate entries in a cross-referencing list.

```
4776 \newcommand*\glsseelastsep{\space\andname\space}
```

`\glsseesep` Separator to use between entires in a cross-referencing list.

```
4777 \newcommand*\glsseesep}{, }
```

`\glsseeitem` `\glsseeitem{<label>}` formats individual entry in a cross-referencing list.

```
4778 \DeclareRobustCommand*\glsseeitem[1]{\gls hyperlink[\glsseeitemformat{#1}]{#1}}
```

`\glsseeitemformat` As from v3.0, default is to use `\glsentrytext` instead of `\glsentryname`. (To avoid problems with the name key being sanitized.)

```
4779 \newcommand*\glsseeitemformat[1]{\glsentrytext{#1}}
```

1.15 Displaying the glossary

An individual glossary is displayed in the text using `\printglossary[<key-val list>]`. If the type key is omitted, the default glossary is displayed. The optional argument can be used to specify an alternative glossary, and can also be used to set the style, title and entry in the table of contents. Available keys are defined below.

`\gls@save@numberlist` Provide command to store number list.

```
4780 \newcommand*\gls@save@numberlist[1]{%
4781   \ifglssavenumberlist
4782     \toks@{#1}%
4783     \edef\@do@writeaux@info{%
4784       \noexpand\csgdef{glo@\glscurrententrylabel @numberlist}{\the\toks@}%
4785     }%
4786     \@onelevel@sanitize\@do@writeaux@info
4787     \protected@write\@auxout{}\@do@writeaux@info}%
4788   \fi
4789 }
```

`\warn@noprintglossary` Warn the user if they have forgotten `\printglossaries` or `\printglossary`. (Will be suppressed if there is at least one occurrence of `\printglossary`. There is no check to ensure that there is a `\printglossary` for each defined glossary.)

```
4790 \newcommand*\warn@noprintglossary}{%}
```

`\printglossary` The TOC title needs to be processed in a different manner to the main title in case the translator and hyperref packages are both being used.

```
4791 \ifcsundef{printglossary}{}%
4792 {%
```

If `\printglossary` is already defined, issue a warning and undefine it.

```
4793   \@gls@warnonglossdefined
4794   \undef\printglossary
4795 }
```

`\printglossary` has an optional argument. The default value is to set the glossary type to the main glossary.

```
4796 \newcommand*\printglossary}[1][type=\glsdefaulttype]{%
4797   \@printglossary{#1}{\@print@glossary}}%
4798 }
```

The `\printglossaries` command will do `\printglossary` for each glossary type that has been defined. It is better to use `\printglossaries` rather than individual `\printglossary` commands to ensure that you don't forget any new glossaries you may have created. It also makes it easier to chop and change the value of the acronym package option. However, if you want to list the glossaries in a different order, or if you want to set the title or table of contents entry, or if you want to use different glossary styles for each glossary, you will need to use `\printglossary` explicitly for each glossary type.

`\printglossaries`

```
4799 \newcommand*\printglossaries{%
4800   \forallglossaries{\@glo@type}{\printglossary[type=\@glo@type]}%
4801 }
```

`\printnoidxglossary` Provide an alternative to `\printglossary` that doesn't require an external indexing application. Entries won't be sorted and the location list will be empty.

```
4802 \newcommand*\printnoidxglossary}[1][type=\glsdefaulttype]{%
4803   \@printglossary{#1}{\@print@noidx@glossary}}%
4804 }
```

`\printnoidxglossaries` Analogous to `\printglossaries`

```
4805 \newcommand*\printnoidxglossaries{%
4806   \forallglossaries{\@glo@type}{\printnoidxglossary[type=\@glo@type]}%
4807 }
```

`\@printgloss@setsort` Initialise to do nothing.

```
4808 \newcommand*\@printgloss@setsort}{}
```

`\@printglossary` Sets up the glossary for either `\printglossary` or `\printnoidxglossary`. The first argument is the options list, the second argument is the handler macro that deals with the actual glossary.

```
4809 \newcommand{\@printglossary}[2]{%
  Set up defaults.
4810   \def\@glo@type{\glsdefaulttype}%
4811   \def\glossarytitle{\csname @glo@type @title\endcsname}%
4812   \def\glossarytoctitle{\glossarytitle}%
4813   \let\org@glossarytitle\glossarytitle
4814   \def\@glossarystyle{}%
4815   \def\gls@dotoc@title{\glssettoctitle{\@glo@type}}%
```

Store current value of `\glossaryentrynumbers`. (This may be changed via the optional argument)

```
4816 \let\org@glossaryentrynumbers\glossaryentrynumbers
```

Localise the effects of the optional argument

```
4817 \bgroup
```

Activate or deactivate sort key:

```
4818 \@printgloss@setsort
```

Determine settings specified in the optional argument.

```
4819 \setkeys{printgloss}{#1}%
```

If title has been set, but `toctitle` hasn't, make `toctitle` the same as given title (rather than the title used when the glossary was defined)

```
4820 \ifx\glossarytitle\org@glossarytitle
4821 \else
4822 \expandafter\let\csname @glo@type @title\endcsname
4823 \glossarytitle
4824 \fi
```

Allow a high-level user command to indicate the current glossary

```
4825 \let\currentglossary\@glo@type
```

Enable individual number lists to be suppressed.

```
4826 \let\org@glossaryentrynumbers\glossaryentrynumbers
4827 \let\glsnonextpages\@glsnonextpages
```

Enable individual number list to be activated:

```
4828 \let\glsnextpages\@glsnextpages
```

Enable suppression of description terminators.

```
4829 \let\nopostdesc\@nopostdesc
```

Set up the entry for the TOC

```
4830 \gls@dotoc@title
```

Set the glossary style

```
4831 \@glossarystyle
```

Added a way to fetch the current entry label (v3.08 updated for new `\glossentry` and `\subglossentry`, but this is now only needed for backward compatibility):

```
4832 \let\gls@org@glossaryentryfield\glossentry
4833 \let\gls@org@glossarysubentryfield\subglossentry
4834 \renewcommand{\glossentry}[1]{%
4835 \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
4836 \gls@org@glossaryentryfield{##1}%
4837 }%
4838 \renewcommand{\subglossentry}[2]{%
4839 \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
4840 \gls@org@glossarysubentryfield{##1}{##2}%
4841 }%
```


Now do the handler macro that deals with the actual glossary:

```
4842   #2%
      End the current scope
4843   \egroup
      Reset \glossaryentrynumbers
4844   \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
      Suppress warning about no \printglossary
4845   \global\let\warn@noprintglossary\relax
4846 }
```

\@print@glossary Internal workings of \printglossary dealing with reading the external file.

```
4847 \newcommand{\@print@glossary}{%
```

Some macros may end up being expanded into internals in the glossary, so need to make @ a letter. (Unlikely to be a problem since v3.08a but kept for backward compatibility.)

```
4848   \makeatletter
```

Input the glossary file, if it exists.

```
4849   \@input@{\jobname.\csname @glo@type@\@glo@type @in\endcsname}%
```

If the glossary file doesn't exist, do \null. (This ensures that the page is shipped out and all write commands are done.) This might produce an empty page, but at this point the document isn't complete, so it shouldn't matter.

```
4850   \IfFileExists{\jobname.\csname @glo@type@\@glo@type @in\endcsname}%
```

```
4851   {}%
```

```
4852   {\null}%
```

If xindy is being used, need to write the language dependent information to the .aux file for makeglossaries.

```
4853   \ifglxindy
```

```
4854     \ifcsundef{@xdy@\@glo@type @language}%
```

```
4855     {%
```

```
4856       \edef\@do@auxoutstuff{%
```

```
4857         \noexpand\AtEndDocument{%
```

If the user removes the glossary package from their document, ensure the next run doesn't throw a load of undefined control sequence errors when the aux file is parsed.

```
4858           \noexpand\immediate\noexpand\write\@auxout{%
```

```
4859             \string\providecommand\string\@xdy@language[2]{}%}
```

```
4860           \noexpand\immediate\noexpand\write\@auxout{%
```

```
4861             \string\@xdy@language{\@glo@type}{\@xdy@main@language}}%
```

```
4862           }%
```

```
4863         }%
```

```
4864       }%
```

```
4865     {%
```

```
4866       \edef\@do@auxoutstuff{%
```

```

4867     \noexpand\AtEndDocument{%
4868         \noexpand\immediate\noexpand\write\@auxout{%
4869             \string\providecommand\string\@xdy\language[2]{}%
4870         \noexpand\immediate\noexpand\write\@auxout{%
4871             \string\@xdy\language{\@glo@type}\csname @xdy@\@glo@type
4872                 @language\endcsname}}%
4873     }%
4874 }%
4875 }%
4876 \do@auxoutstuff
4877 \edef\do@auxoutstuff{%
4878     \noexpand\AtEndDocument{%

```

If the user removes the glossaries package from their document, ensure the next run doesn't throw a load of undefined control sequence errors when the aux file is parsed.

```

4879     \noexpand\immediate\noexpand\write\@auxout{%
4880         \string\providecommand\string\@gls@codepage[2]{}%
4881     \noexpand\immediate\noexpand\write\@auxout{%
4882         \string\@gls@codepage{\@glo@type}\@gls@codepage}}%
4883     }%
4884 }%
4885 \do@auxoutstuff
4886 \fi

```

Activate warning if `\makeglossaries` hasn't been used.

```

4887 \renewcommand*\@warn@nomakeglossaries{%
4888     \GlossariesWarningNoLine{\string\makeglossaries\space
4889     hasn't been used,^^Jthe glossaries will not be updated}%
4890 }%
4891 }

```

The sort macros all have the syntax:

```
\@glo@sortmacro@<order>{<type>}
```

where `<order>` is the sort order as specified by the sort key and `<type>` is the glossary type. (The referenced entry list is stored in `\@glsref@<type>`). The actual sorting is done by `\@glo@sortentries{<handler>}{<type>}`.

`\@glo@sortentries`

```

4892 \newcommand*\@glo@sortentries}[2]{%
4893     \def\@glo@sortinglist{}%
4894     \def\@glo@sortinghandler{#1}%
4895     \edef\@glo@type{#2}%
4896     \forlistcloop{\@glo@do@sortentries}{\@glsref@#2}%
4897     \csdef{\@glsref@#2}{}%
4898     \@for\@this@label:=\@glo@sortinglist\do{%

```

Has this entry already been added?

```
4899 \xifinlistcs{\@this@label}{@glsref@#2}%
4900 {}%
4901 {%
4902 \listcsxadd{@glsref@#2}{\@this@label}%
4903 }%
4904 \ifcsdef{@glo@sortingchildren@\@this@label}%
4905 {%
4906 \@glo@addchildren{#2}{\@this@label}%
4907 }%
4908 {}%
4909 }%
4910 }
```

`\@glo@addchildren` `\@glo@addchildren{<type>}{<parent>}`

```
4911 \newcommand*{\@glo@addchildren}[2]{%
```

Scope to allow nesting.

```
4912 \bgroup
4913 \letcs{\@glo@childlist}{@glo@sortingchildren@#2}%
4914 \@for\@this@childlabel:=\@glo@childlist\do
4915 {%
```

Check this label hasn't already been added.

```
4916 \xifinlistcs{\@this@childlabel}{@glsref@#1}%
4917 {}%
4918 {%
4919 \listcsxadd{@glsref@#1}{\@this@childlabel}%
4920 }%
```

Does this child have children?

```
4921 \ifcsdef{@glo@sortingchildren@\@this@childlabel}%
4922 {%
4923 \@glo@addchildren{#1}{\@this@childlabel}%
4924 }%
4925 {%
4926 }%
4927 }%
4928 \egroup
4929 }
```

`@glo@do@sortentries`

```
4930 \newcommand*{\@glo@do@sortentries}[1]{%
4931 \ifglshasparent{#1}%
4932 {%
```

This entry has a parent, so add it to the child list

```
4933 \edef\@glo@parent{\csuse{glo@glstetoklabel{#1}@parent}}%
```

```

4934 \ifcsundef{@glo@sortingchildren@\@glo@parent}%
4935 {%
4936 \csdef{@glo@sortingchildren@\@glo@parent}{}%
4937 }%
4938 {}%
4939 \expandafter\@glo@sortedinsert
4940 \csname @glo@sortingchildren@\@glo@parent\endcsname{#1}%

```

Has the parent been added?

```

4941 \xifinlistcs{\@glo@parent}{\@glsref@\@glo@type}%
4942 {%

```

Yes, it has so do nothing.

```

4943 }%
4944 {%

```

No, it hasn't so add it now.

```

4945 \expandafter\@glo@do@sortentries\expandafter{\@glo@parent}%
4946 }%
4947 }%
4948 {}%
4949 \@glo@sortedinsert{\@glo@sortinglist}{#1}%
4950 }%
4951 }

```

`\@glo@sortedinsert` `\@glo@sortedinsert{<list>}{<entry label>}`

Insert into list.

```

4952 \newcommand*{\@glo@sortedinsert}[2]{%
4953 \dtl@insertinto{#2}{#1}{\@glo@sortinghandler}%
4954 }%

```

The sort handlers need to be in the form required by datatool's `\dtl@sortlist` macro. These must set the count register `\dtl@sortresult` to either `-1` (`#1` less than `#2`), `0` (`#1 = #2`) or `+1` (`#1` greater than `#2`).

`\@glo@sorthandler@word`

```

4955 \newcommand*{\@glo@sorthandler@word}[2]{%
4956 \letcs\@gls@sort@A{glo@\@glsdetoklabel{#1}@sort}%
4957 \letcs\@gls@sort@B{glo@\@glsdetoklabel{#2}@sort}%
4958 \edef\@glo@do@compare{%
4959 \noexpand\dtlwordindexcompare{\noexpand\dtl@sortresult}%
4960 {\expandonce\@gls@sort@B}%
4961 {\expandonce\@gls@sort@A}%
4962 }%
4963 \@glo@do@compare
4964 }

```

@sorthandler@letter

```
4965 \newcommand*{\@glo@sorthandler@letter}[2]{%
4966   \letcs\@gls@sort@A{glo@glstetoklabel{#1}@sort}%
4967   \letcs\@gls@sort@B{glo@glstetoklabel{#2}@sort}%
4968   \edef\glo@do@compare{%
4969     \noexpand\dtlletterindexcompare{\noexpand\dtl@sortresult}%
4970     {\expandonce\@gls@sort@B}%
4971     {\expandonce\@gls@sort@A}%
4972   }%
4973   \glo@do@compare
4974 }
```

l@sorthandler@case Case-sensitive sort.

```
4975 \newcommand*{\@glo@sorthandler@case}[2]{%
4976   \letcs\@gls@sort@A{glo@glstetoklabel{#1}@sort}%
4977   \letcs\@gls@sort@B{glo@glstetoklabel{#2}@sort}%
4978   \edef\glo@do@compare{%
4979     \noexpand\dtlcompare{\noexpand\dtl@sortresult}%
4980     {\expandonce\@gls@sort@B}%
4981     {\expandonce\@gls@sort@A}%
4982   }%
4983   \glo@do@compare
4984 }
```

@sorthandler@nocase Case-insensitive sort.

```
4985 \newcommand*{\@glo@sorthandler@nocase}[2]{%
4986   \letcs\@gls@sort@A{glo@glstetoklabel{#1}@sort}%
4987   \letcs\@gls@sort@B{glo@glstetoklabel{#2}@sort}%
4988   \edef\glo@do@compare{%
4989     \noexpand\dtlicompare{\noexpand\dtl@sortresult}%
4990     {\expandonce\@gls@sort@B}%
4991     {\expandonce\@gls@sort@A}%
4992   }%
4993   \glo@do@compare
4994 }
```

@glo@sortmacro@word Sort macro for ‘word’

```
4995 \newcommand*{\@glo@sortmacro@word}[1]{%
4996   \ifdefstring{\@glo@default@sorttype}{standard}%
4997   {%
4998     \@glo@sortentries{\@glo@sorthandler@word}{#1}%
4999   }%
5000   {%
5001     \PackageError{glossaries}{Conflicting sort options:^^J
5002     \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5003     \string\printnoidxglossary[sort=word]}{}%
5004   }%
5005 }
```

lo@sortmacro@letter Sort macro for 'letter'

```
5006 \newcommand*\@glo@sortmacro@letter}[1]{%
5007   \ifdefstring{\@glo@default@sorttype}{standard}%
5008   {%
5009     \@glo@sortentries{\@glo@sorthandler@letter}{#1}%
5010   }%
5011   {%
5012     \PackageError{glossaries}{Conflicting sort options:^^J
5013       \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5014       \string\printnoidxglossary[sort=letter]}{}%
5015   }%
5016 }
```

@sortmacro@standard Sort macro for 'standard'. (Use either 'word' or 'letter' order.)

```
5017 \newcommand*\@glo@sortmacro@standard}[1]{%
5018   \ifdefstring{\@glo@default@sorttype}{standard}%
5019   {%
5020     \ifcsdef{@glo@sorthandler@\glsorder}%
5021     {%
5022       \@glo@sortentries{\csuse{@glo@sorthandler@\glsorder}}{#1}%
5023     }%
5024     {%
5025       \PackageError{glossaries}{Unknown sort handler '\glsorder'}{}%
5026     }%
5027   }%
5028   {%
5029     \PackageError{glossaries}{Conflicting sort options:^^J
5030       \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5031       \string\printnoidxglossary[sort=standard]}{}%
5032   }%
5033 }
```

@glo@sortmacro@case Sort macro for 'case'

```
5034 \newcommand*\@glo@sortmacro@case}[1]{%
5035   \ifdefstring{\@glo@default@sorttype}{standard}%
5036   {%
5037     \@glo@sortentries{\@glo@sorthandler@case}{#1}%
5038   }%
5039   {%
5040     \PackageError{glossaries}{Conflicting sort options:^^J
5041       \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5042       \string\printnoidxglossary[sort=case]}{}%
5043   }%
5044 }
```

lo@sortmacro@nocase Sort macro for 'nocase'

```
5045 \newcommand*\@glo@sortmacro@nocase}[1]{%
5046   \ifdefstring{\@glo@default@sorttype}{standard}%
5047   {%
```

```

5048   \@glo@sortentries{\@glo@sorthandler@nocase}{#1}%
5049 }%
5050 {%
5051   \PackageError{glossaries}{Conflicting sort options:^^J
5052   \string\usepackage[sort=\@glo@default@sorttype]{glossaries}^^J
5053   \string\printnoidxglossary[sort=nocase]}{}%
5054 }%
5055 }

```

`\@glo@sortmacro@def` Sort macro for ‘def’. The order of definition is given in `\glo@list@<type>`.

```

5056 \newcommand*\@glo@sortmacro@def}[1]{%
5057   \def\@glo@sortinglist{}%
5058   \for@gl@entries[#1]{\@gl@thislabel}%
5059   {%
5060     \xifinlistcs{\@gl@thislabel}{\@gl@sref@#1}%
5061     {%
5062       \list@add{\@glo@sortinglist}{\@gl@thislabel}%
5063     }%
5064     {%
5065       Hasn't been referenced.
5066     }%
5067     \cslet{\@gl@sref@#1}{\@glo@sortinglist}%
5068   }

```

`\@glo@sortmacro@def@do` This won't include parent entries that haven't been referenced.

```

5069 \newcommand*\@glo@sortmacro@def@do}[1]{%
5070   \ifinlistcs{#1}{\@gl@sref@\@glo@type}%
5071   {}%
5072   {%
5073     \list@csadd{\@gl@sref@\@glo@type}{#1}%
5074   }%
5075   \ifcsdef{\@glo@sortingchildren@#1}%
5076   {%
5077     \@glo@addchildren{\@glo@type}{#1}%
5078   }%
5079   {}%
5080 }

```

`\@glo@sortmacro@use` Sort macro for ‘use’. (No sorting is required, as the entries are already in order of use, so do nothing.)

```

5081 \newcommand*\@glo@sortmacro@use}[1]{%

```

`\printnoidx@glossary` Glossary handler for `\printnoidxglossary` which doesn't use an indexing application. Since `\printnoidxglossary` may occur at the start of the document, we can't just check if an entry has been used. Instead, the first pass needs to write information to the aux file every time an entry is referenced. This needs

to be read in on the second run and stored in a list corresponding to the appropriate glossary.

```
5082 \newcommand*{\@print@noidx@glossary}{%
5083   \ifcsdef{@glsref@\@glo@type}%
5084   {%
```

Sort the entries:

```
5085   \ifcsdef{@glo@sortmacro@\@glo@sorttype}%
5086   {%
5087     \csuse{@glo@sortmacro@\@glo@sorttype}{\@glo@type}%
5088   }%
5089   {%
5090     \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
5091   }%
```

Do the glossary heading and preamble

```
5092   \glossarysection[\glossarytoctitle]{\glossarytitle}%
5093   \glossarypreamble
5094   \begin{theglossary}%
5095   \glossaryheader
5096   \glsresetentrylist
5097   \def\@gls@currentlettergroup{}
```

Iterate through the entries.

```
5098   \forlistcsloop{\@gls@noidx@do}{\@glsref@\@glo@type}%
```

Finally end the glossary and do the postamble:

```
5099   \end{theglossary}%
5100   \glossarypostamble
5101 }%
5102 {%
5103   \@gls@noref@warn{\@glo@type}%
5104 }%
5105 }
```

\glo@grabfirst

```
5106 \def\glo@grabfirst#1#2\@nil{%
5107   \def\@gls@firsttok{#1}%
5108   \ifdefempty\@gls@firsttok
5109   {%
5110     \def\@glo@thislettergrp{0}%
5111   }%
5112   {%
```

Sanitize it:

```
5113   \@onelevel@sanitize\@gls@firsttok
```

Fetch the first letter:

```
5114   \expandafter\@glo@grabfirst\@gls@firsttok{}{}\@nil
5115 }%
5116 }
```


`\@glo@grabfirst`

```
5117 \def\@glo@grabfirst#1#2\@nil{%
5118   \ifdefempty\@glo@thislettergrp
5119   {%
5120     \def\@glo@thislettergrp{glssymbols}%
5121   }%
5122   {%
5123     \count@=\ucode‘#1\relax
5124     \ifnum\count@=0\relax
5125       \def\@glo@thislettergrp{glssymbols}%
5126     \else
5127       \ifdefstring\@glo@sorttype{case}%
5128       {%
5129         \count@=‘#1\relax
5130       }%
5131     }%
5132   }%
5133   \edef\@glo@thislettergrp{\the\count@}%
5134   \fi
5135 }%
5136 }
```

`\@gls@noidx@do` Handler for list iteration used by `\@print@noidx@glossary`. The argument is the entry label. This only allows one sublevel.

```
5137 \newcommand{\@gls@noidx@do}[1]{%
```

Get this entry's location list

```
5138   \global\letcs{\@gls@loclist}{glo@glstdetoklabel{#1}@loclist}%
```

Does this entry have a parent?

```
5139   \ifglshasparent{#1}%
5140   {%
```

Has a parent.

```
5141     \gls@level=\csuse{glo@glstdetoklabel{#1}@level}\relax
5142     \ifdefvoid{\@gls@loclist}
5143     {%
5144       \subglossentry{\@gls@level}{#1}{}%
5145     }%
5146     {%
5147       \subglossentry{\@gls@level}{#1}%
5148     }%
5149     \glossaryentrynumbers{\@gls@loclist{\@gls@loclist}}%
5150   }%
5151 }%
5152 }%
5153 {%
```

Doesn't have a parent Get this entry's sort key

```
5154   \letcs{\@gls@sort}{glo@glstdetoklabel{#1}@sort}%
```

Fetch the first letter:

```
5155 \expandafter\glo@grabfirst\@gls@sort{}{} \@nil
5156 \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
5157 {}%
5158 {}%
```

Do the group header:

```
5159 \ifdefempty{\@gls@currentlettergroup}{\@gls@groupskip}%
5160 \gls@groupheading{\@glo@thislettergrp}%
5161 }%
5162 \let\@gls@currentlettergroup\@glo@thislettergrp
```

Do this entry:

```
5163 \ifdefvoid{\@gls@loclist}
5164 {}%
5165 \glossentry{#1}{}%
5166 }%
5167 {}%
5168 \glossentry{#1}%
5169 {}%
5170 \glossaryentrynumbers{\@gls@noidxloclist{\@gls@loclist}}%
5171 }%
5172 }%
5173 }%
5174 }
```

`\glsnoidxloclist` `\glsnoidxloclist{<list cs>}`

Display location list.

```
5175 \newcommand*{\glsnoidxloclist}[1]{}%
5176 \def\@gls@noidxloclist@sep{}%
5177 \def\@gls@noidxloclist@prev{}%
5178 \forlistloop{\glsnoidxloclisthandler}{#1}%
5179 }
```

`noidxloclisthandler` Handler for location list iterator.

```
5180 \newcommand*{\glsnoidxloclisthandler}[1]{}%
5181 \ifdefstring{\@gls@noidxloclist@prev}{#1}%
5182 {}%
```

Same as previous location so skip.

```
5183 }%
5184 {}%
5185 \@gls@noidxloclist@sep
5186 #1%
5187 \def\@gls@noidxloclist@sep{\delimN}%
5188 \def\@gls@noidxloclist@prev{#1}%
5189 }%
5190 }
```

`\displayloclisthandler` Handler for location list iterator when used with `\glsdisplaynumberlist`.

```
5191 \newcommand*\glsnoidxdisplayloclisthandler}[1]{%
5192   \ifdefstring{\@gls@noidxloclist@prev}{#1}%
5193   {%
      Same as previous location so skip.
5194   }%
5195   {%
5196     \@gls@noidxloclist@sep
5197     \@gls@noidxloclist@prev
5198     \def\@gls@noidxloclist@prev{#1}%
5199   }%
5200 }
```

`\glsnoidxdisplayloc` `\glsnoidxdisplayloc{<prefix>}{<counter>}{<format>}{<location>}`

Display a location in the location list.

```
5201 \newcommand*\glsnoidxdisplayloc[4]{%
5202   \setentrycounter[#1]{#2}%
5203   \csuse{#3}{#4}%
5204 }
```

`\@gls@reference` `\@gls@reference{<type>}{<label>}{<loc>}`

Identifies that a reference has been used (for use in the aux file). All entries must be defined in the preamble.

```
5205 \newcommand*\@gls@reference}[3]{%
      Add to label list
5206   \glsdoifexistsorwarn{#2}%
5207   {%
5208     \ifcsundef{@glsref@#1}{\csgdef{@glsref@#1}{}}{}%
5209     \ifinlistcs{#2}{@glsref@#1}%
5210     {}%
5211     {\listcsgadd{@glsref@#1}{#2}}%
      Add to location list
5212   \ifcsundef{glo@glstdetoklabel{#2}@loclist}%
5213   {\csgdef{glo@glstdetoklabel{#2}@loclist}{}}%
5214   {}%
5215   \listcsgadd{glo@glstdetoklabel{#2}@loclist}{#3}%
5216   }%
5217 }
```

The keys that can be used in the optional argument to `\printglossary` or `\printnoidxglossary` are as follows: The type key sets the glossary type.

```
5218 \define@key{printgloss}{type}{\def\@glo@type{#1}}
```

The title key sets the title used in the glossary section header. This overrides the title used in `\newglossary`.

```
5219 \define@key{printgloss}{title}{%
5220 \def\glossarytitle{#1}%
5221 \let\gls@dotoc\title\relax
5222 }
```

The toctitle sets the text used for the relevant entry in the table of contents.

```
5223 \define@key{printgloss}{toctitle}{%
5224 \def\glossarytoctitle{#1}%
5225 \let\gls@dotoc\toctitle\relax
5226 }
```

The style key sets the glossary style (but only for the given glossary).

```
5227 \define@key{printgloss}{style}{%
5228 \ifcsundef{@glsstyle@#1}%
5229 {%
5230 \PackageError{glossaries}%
5231 {Glossary style ‘#1’ undefined}{}%
5232 }%
5233 {%
5234 \def\@glossarystyle{\setglossentrycompatibility
5235 \csname @glsstyle@#1\endcsname}%
5236 }%
5237 }
```

The numberedsection key determines if this glossary should be in a numbered section.

```
5238 \define@choicekey{printgloss}{numberedsection}[\val\nr]{%
5239 false,nolabel,autolabel,nameref}[nolabel]{%
5240 \ifcase\nr\relax
5241 \renewcommand*{\@glossarysecstar}{*}%
5242 \renewcommand*{\@glossaryseclabel}{}%
5243 \or
5244 \renewcommand*{\@glossarysecstar}{}%
5245 \renewcommand*{\@glossaryseclabel}{}%
5246 \or
5247 \renewcommand*{\@glossarysecstar}{*}%
5248 \renewcommand*{\@glossaryseclabel}{\label{\glsautoprefix\@glo@type}}%
5249 \or
5250 \renewcommand*{\@glossarysecstar}{*}%
5251 \renewcommand*{\@glossaryseclabel}{%
5252 \protected@edef\@currentlabelname{\glossarytoctitle}%
5253 \label{\glsautoprefix\@glo@type}}%
5254 \fi
5255 }
```

The nogroupskip key determines whether or not there should be a vertical gap between glossary groups.

```
5256 \define@choicekey{printgloss}{nogroupskip}{true,false}[true]{%
```

```
5257 \csuse{glsnogroupskip#1}%
5258 }
```

The nopostdot key has the same effect as the package option of the same name.

```
5259 \define@choicekey{printgloss}{nopostdot}{true,false}[true]{%
5260 \csuse{glsnopostdot#1}%
5261 }
```

The entrycounter key is the same as the package option but localised to the current glossary.

```
5262 \define@choicekey{printgloss}{entrycounter}{true,false}[true]{%
5263 \csuse{glsentrycounter#1}%
5264 \ifglsentrycounter
5265 \ifx\@gls@counterwithin\@empty
5266 \newcounter{glossaryentry}%
5267 \else
5268 \newcounter{glossaryentry}[\@gls@counterwithin]%
5269 \fi
5270 \def\theHglossaryentry{\currentglossary.\theglossaryentry}%
5271 \renewcommand*{\glsresetentrycounter}{%
5272 \setcounter{glossaryentry}{0}%
5273 }%
5274 \renewcommand*{\glsstepentry}[1]{%
5275 \refstepcounter{glossaryentry}%
5276 \label{glsentry-\glsdetoklabel{##1}}%
5277 }%
5278 \renewcommand*{\glsentrycounterlabel}{\theglossaryentry.\space}%
5279 \renewcommand*{\glsentryitem}[1]{%
5280 \glsstepentry{##1}\glsentrycounterlabel
5281 }%
5282 \else
5283 \renewcommand*{\glsresetentrycounter}{}%
5284 \renewcommand*{\glsstepentry}[1]{%
5285 \renewcommand*{\glsentrycounterlabel}{}%
5286 \renewcommand*{\glsentryitem}[1]{\glsresetsubentrycounter}
5287 \fi
5288 }
```

The subentrycounter key is the same as the package option but localised to the current glossary. Note that this doesn't affect the master/slave counter attributes, which occurs if subentrycounter and entrycounter package options are set to true.

```
5289 \define@choicekey{printgloss}{subentrycounter}{true,false}[true]{%
5290 \csuse{glssubentrycounter#1}%
5291 \ifglssubentrycounter
5292 \ifundef\c@glossarysubentry
5293 {%
5294 \ifglsentrycounter
5295 \newcounter{glossarysubentry}[glossaryentry]%

```

```

5296     \else
5297         \newcounter{glossarysubentry}
5298     \fi
5299 }{}%
5300 \renewcommand*{\glsstepsubentry}[1]{%
5301     \edef\currentglsentry{\glsdetoklabel{##1}}%
5302     \refstepcounter{glossarysubentry}%
5303     \label{glsentry-\currentglsentry}%
5304 }%
5305 \renewcommand*{\glsresetsubentrycounter}{%
5306     \setcounter{glossarysubentry}{0}%
5307 }%
5308 \renewcommand*{\glsentryitem}[1]{%
5309     \glsstepsubentry{##1}\glsentrycounterlabel
5310 }%
5311 \renewcommand*{\glsentrycounterlabel}{\theglossarysubentry}\space}%
5312 \def\theHglossarysubentry{\currentglsentry.\theglossarysubentry}
5313 \else
5314     \renewcommand*{\glsentryitem}[1]{}%
5315     \renewcommand*{\glsstepsubentry}[1]{}%
5316     \renewcommand*{\glsresetsubentrycounter}{}%
5317     \renewcommand*{\glsentrycounterlabel}{}%
5318 \fi
5319 }

```

The nonnumberlist key determines if this glossary should have a number list.

```

5320 \define@boolkey{printgloss}[gls]{nonnumberlist}[true]{}%
5321 \ifglsnonnumberlist
5322     \def\glossaryentrynumbers##1{}%
5323 \else
5324     \def\glossaryentrynumbers##1{##1}%
5325 \fi}

```

The sort key sets the glossary sort handler (`\printnoidxglossary` only).

```

5326 \define@key{printgloss}{sort}{\@glo@assign@sortkey{#1}}

```

`\@glo@no@assign@sortkey` Issue error if used with `\printglossary`

```

5327 \newcommand*{\@glo@no@assign@sortkey}[1]{}%
5328 \PackageError{glossaries}{‘sort’ key not permitted with
5329 \string\printglossary}%
5330 {The ‘sort’ key may only be used with \string\printnoidxglossary}%
5331 }

```

`\@glo@assign@sortkey` For use with `\printnoidxglossary`

```

5332 \newcommand*{\@glo@assign@sortkey}[1]{}%
5333 \def\@glo@sorttype{#1}%
5334 }

```

`\@glsnonextpages` Suppresses the next number list only. Global assignments required as it may not occur in the same level of grouping as the next numberlist. (For example, if

`\glsnonextpages` is place in the entry's description and 3 column tabular style glossary is used.) `\org@glossaryentrynumbers` needs to be set at the start of each glossary, in the event that `\glossaryentrynumber` is redefined.

```
5335 \newcommand*{\@glsnonextpages}{%
5336   \gdef\glossaryentrynumbers##1{%
5337     \glsresetentrylist
5338   }%
5339 }
```

`\@glsnextpages` Activate the next number list only. Global assignments required as it may not occur in the same level of grouping as the next numberlist. (For example, if `\glsnextpages` is place in the entry's description and 3 column tabular style glossary is used.) `\org@glossaryentrynumbers` needs to be set at the start of each glossary, in the event that `\glossaryentrynumber` is redefined.

```
5340 \newcommand*{\@glsnextpages}{%
5341   \gdef\glossaryentrynumbers##1{%
5342     ##1\glsresetentrylist}}
```

`\glsresetentrylist` Resets `\glossaryentrynumbers`

```
5343 \newcommand*{\glsresetentrylist}{%
5344   \global\let\glossaryentrynumbers\org@glossaryentrynumbers}
```

`\glsnonextpages` Outside of `\printglossary` this does nothing.

```
5345 \newcommand*{\glsnonextpages}{}
```

`\glsnextpages` Outside of `\printglossary` this does nothing.

```
5346 \newcommand*{\glsnextpages}{}
```

`glossaryentry` If the `entrycounter` package option has been used, define a counter to number each level 0 entry.

```
5347 \ifglentrycounter
5348   \ifx\@gls@counterwithin\@empty
5349     \newcounter{glossaryentry}
5350   \else
5351     \newcounter{glossaryentry}[\@gls@counterwithin]
5352   \fi
5353   \def\theHglossaryentry{\currentglossary.\theglossaryentry}
5354 \fi
```

`glossarysubentry` If the `subentrycounter` package option has been used, define a counter to number each level 1 entry.

```
5355 \ifglssubentrycounter
5356   \ifglentrycounter
5357     \newcounter{glossarysubentry}[glossaryentry]
5358   \else
5359     \newcounter{glossarysubentry}
5360   \fi
```

```

5361 \def\theHglossarysubentry{\currentglssubentry.\theglossarysubentry}
5362 \fi

```

`\glsresetsubentrycounter` Resets the glossarysubentry counter.

```

5363 \ifglssubentrycounter
5364 \newcommand*\glsresetsubentrycounter{%
5365 \setcounter{glossarysubentry}{0}%
5366 }
5367 \else
5368 \newcommand*\glsresetsubentrycounter{}
5369 \fi

```

`\glsresetentrycounter` Resets the glossentry counter.

```

5370 \ifglentrycounter
5371 \newcommand*\glsresetentrycounter{%
5372 \setcounter{glossaryentry}{0}%
5373 }
5374 \else
5375 \newcommand*\glsresetentrycounter{}
5376 \fi

```

`\glsstepentry` Advance the glossaryentry counter if in use. The argument is the label associated with the entry.

```

5377 \ifglentrycounter
5378 \newcommand*\glsstepentry}[1]{%
5379 \refstepcounter{glossaryentry}%
5380 \label{glsentry-\glsdetoklabel{#1}}%
5381 }
5382 \else
5383 \newcommand*\glsstepentry}[1]{}
5384 \fi

```

`\glsstepsubentry` Advance the glossarysubentry counter if in use. The argument is the label associated with the subentry.

```

5385 \ifglssubentrycounter
5386 \newcommand*\glsstepsubentry}[1]{%
5387 \edef\currentglssubentry{\glsdetoklabel{#1}}%
5388 \refstepcounter{glossarysubentry}%
5389 \label{glsentry-\currentglssubentry}%
5390 }
5391 \else
5392 \newcommand*\glsstepsubentry}[1]{}
5393 \fi

```

`\glsrefentry` Reference the entry or sub-entry counter if in use, otherwise just do `\gls`.

```

5394 \ifglentrycounter
5395 \newcommand*\glsrefentry}[1]{\ref{glsentry-\glsdetoklabel{#1}}}
5396 \else

```



```

5397 \ifglssubentrycounter
5398   \newcommand*{\glsrefentry}[1]{\ref{glsentry-\glsdetoklabel{#1}}}
5399 \else
5400   \newcommand*{\glsrefentry}[1]{\gls{#1}}
5401 \fi
5402 \fi

```

`glsentrycounterlabel` Defines how to display the glossaryentry counter.

```

5403 \ifglssentrycounter
5404   \newcommand*{\glsentrycounterlabel}{\theglossaryentry.\space}
5405 \else
5406   \newcommand*{\glsentrycounterlabel}{}
5407 \fi

```

`glssubentrycounterlabel` Defines how to display the glossarysubentry counter.

```

5408 \ifglssubentrycounter
5409   \newcommand*{\glssubentrycounterlabel}{\theglossarysubentry)\space}
5410 \else
5411   \newcommand*{\glssubentrycounterlabel}{}
5412 \fi

```

`glsentryitem` Step and display glossaryentry counter, if appropriate.

```

5413 \ifglssentrycounter
5414   \newcommand*{\glsentryitem}[1]{%
5415     \glsstepentry{#1}\glsentrycounterlabel
5416   }
5417 \else
5418   \newcommand*{\glsentryitem}[1]{\glsresetsubentrycounter}
5419 \fi

```

`glssubentryitem` Step and display glossarysubentry counter, if appropriate.

```

5420 \ifglsssubentrycounter
5421   \newcommand*{\glssubentryitem}[1]{%
5422     \glsstepsubentry{#1}\glssubentrycounterlabel
5423   }
5424 \else
5425   \newcommand*{\glssubentryitem}[1]{}
5426 \fi

```

`theglossary` If the `theglossary` environment has already been defined, a warning will be issued. This environment should be redefined by glossary styles.

```

5427 \ifcsundef{theglossary}%
5428 {%
5429   \newenvironment{theglossary}{}{}%
5430 }%
5431 {%
5432   \@gls@warnontheglossdefined
5433   \renewenvironment{theglossary}{}{}%
5434 }

```

The glossary header is given by `\glossaryheader`. This forms part of the glossary style, and must indicate what should appear immediately after the start of the `theglossary` environment. (For example, if the glossary uses a tabular-like environment, it may be used to set the header row.) Note that if you don't want a header row, the glossary style must redefine `\glossaryheader` to do nothing.

`\glossaryheader`

```
5435 \newcommand*\glossaryheader{}
```

`\glstarget` `\glstarget{<label>}{<name>}`

Provide user interface to `\@glstarget` to make it easier to modify the glossary style in the document.

```
5436 \newcommand*\glstarget}[2]{\@glstarget{\glo@linkprefix#1}{#2}}
```

As from version 3.08, glossary information is now written to the external files using `\glossentry` and `\subglossentry` instead of `\glossaryentryfield` and `\glossarysubentryfield`. The default definition provides backward compatibility for glossary styles that use the old forms.

`compatibleglossentry`

```
\glossentry{<label>}{<page-list>}
```

```
5437 \providecommand*\compatibleglossentry}[2]{%
5438   \toks@{#2}%
5439   \protected@edef\@do@glossentry{\noexpand\glossaryentryfield{#1}%
5440     {\noexpand\glsnamefont
5441       {\expandafter\expandonce\csname glo@#1@name\endcsname}}%
5442     {\expandafter\expandonce\csname glo@#1@desc\endcsname}%
5443     {\expandafter\expandonce\csname glo@#1@symbol\endcsname}%
5444     {\the\toks@}}%
5445   }%
5446   \@do@glossentry
5447 }
```

`\glossentryname`

```
5448 \newcommand*\glossentryname}[1]{%
5449   \glsdoifexistsorwarn{#1}%
5450   {%
5451     \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5452     \expandafter\glsnamefont\expandafter{\glo@name}%
5453   }%
5454 }
```

`\Glossentryname`

```
5455 \newcommand*\Glossentryname}[1]{%
```

```

5456 \glsdoifexistsorwarn{#1}%
5457 {%
5458     \glsnamefont{\Glsentryname{#1}}%
5459 }%
5460 }

```

`\glossentrydesc`

```

5461 \newcommand*{\glossentrydesc}[1]{%
5462     \glsdoifexistsorwarn{#1}%
5463     {%
5464         \glsentrydesc{#1}%
5465     }%
5466 }

```

`\Glossentrydesc`

```

5467 \newcommand*{\Glossentrydesc}[1]{%
5468     \glsdoifexistsorwarn{#1}%
5469     {%
5470         \Glsentrydesc{#1}%
5471     }%
5472 }

```

`\glossentrysymbol`

```

5473 \newcommand*{\glossentrysymbol}[1]{%
5474     \glsdoifexistsorwarn{#1}%
5475     {%
5476         \glsentrysymbol{#1}%
5477     }%
5478 }

```

`\Glossentrysymbol`

```

5479 \newcommand*{\Glossentrysymbol}[1]{%
5480     \glsdoifexistsorwarn{#1}%
5481     {%
5482         \Glsentrysymbol{#1}%
5483     }%
5484 }

```

`compatiblesubglossentry`

```
\subglossentry{<level>}{<label>}{<page-list>}
```

```

5485 \providecommand*{\compatiblesubglossentry}[3]{%
5486     \toks@{#3}%
5487     \protected@edef\@do@subglossentry{\noexpand\glossarysubentryfield{\number#1}%
5488     {#2}%
5489     {\noexpand\glsnamefont
5490     {\expandafter\expandonce\csname glo@#2@name\endcsname}}%
5491     {\expandafter\expandonce\csname glo@#2@desc\endcsname}%

```

```

5492   {\expandafter\expandonce\csname glo@#2@symbol\endcsname}%
5493   {\the\toks@}%
5494   }%
5495   \@do@subglossentry
5496 }

```

sentrycompatibility

```

5497 \newcommand*\setglossentrycompatibility{%
5498   \let\glossentry\compatibleglossentry
5499   \let\subglossentry\compatiblesubglossentry
5500 }
5501 \setglossentrycompatibility

```

\glossaryentryfield

```
\glossaryentryfield{<label>}{<name>}{<description>}{<symbol>}{<page-list>}
```

This command formerly governed how each entry row should be formatted in the glossary. Now deprecated.

```

5502 \newcommand{\glossaryentryfield}[5]{%
5503   \GlossariesWarning
5504   {Deprecated use of \string\glossaryentryfield.^^J
5505     I recommend you change to \string\glossentry.^^J
5506     If you've just upgraded, try removing your gls auxiliary
5507     files^^J and recompile}%
5508   \noindent\textbf{\glstarget{#1}{#2}} #4 #3. #5\par}

```

lossarysubentryfield

```
\glossarysubentryfield{<level>}{<label>}{<name>}{<description>}{<symbol>}{<page-list>}
```

This command governs how each subentry should be formatted in the glossary. Glossary styles need to redefine this command. Most of the predefined styles ignore *<symbol>*. The first argument is a number indicating the level. (The level should be greater than or equal to 1.)

```

5509 \newcommand*\glossarysubentryfield}[6]{%
5510   \GlossariesWarning
5511   {Deprecated use of \string\glossarysubentryfield.^^J
5512     I recommend you change to \string\subglossentry.^^J
5513     If you've just upgraded, try removing your gls auxiliary
5514     files^^J and recompile}%
5515   \glstarget{#2}{\strut}#4. #6\par}

```

Within each glossary, the entries form distinct groups which are determined by the first character of the sort key. When using `makeindex`, there will be a maximum of 28 groups: symbols, numbers, and the 26 alphabetical groups A, ..., Z. If you use `xindy` the groups will depend on whatever alphabet is

used. This is determined by the language or custom alphabets can be created in the `xindy` style file. The command `\glsgroupskip` specifies what to do between glossary groups. Glossary styles must redefine this command. (Note that `\glsgroupskip` only occurs between groups, not at the start or end of the glossary.)

`\glsgroupskip`

```
5516 \newcommand*{\glsgroupskip}{}
```

Each of the 28 glossary groups described above is preceded by a group heading. This is formatted by the command `\glsgroupheading` which takes one argument which is the *label* assigned to that group (not the title). The corresponding labels are: `glsymbols`, `glsnumbers`, `A`, ..., `Z`. Glossary styles must redefine this command. (In between groups, `\glsgroupheading` comes immediately after `\glsgroupskip`.)

`\glsgroupheading`

```
5517 \newcommand*{\glsgroupheading}[1]{} 
```

It is possible to “trick” `makeindex` into treating entries as though they belong to the same group, even if the terms don’t start with the same letter, by modifying the sort key. For example, all entries belonging to one group could be defined so that the sort key starts with an `a`, while entries belonging to another group could be defined so that the sort key starts with a `b`, and so on. If you want each group to have a heading, you would then need to modify the translation control sequences `\glsgetgrouptitle` and `\glsgetgrouplabel` so that the label is translated into the required title (and vice-versa).

```
\glsgetgrouptitle{<label>}
```

This command produces the title for the glossary group whose label is given by `<label>`. By default, the group labelled `glsymbols` produces `\glsymbolsgroupname`, the group labelled `glsnumbers` produces `\glsnumbersgroupname` and all the other groups simply produce their label. As mentioned above, the group labels are: `glsymbols`, `glsnumbers`, `A`, ..., `Z`. If you want to redefine the group titles, you will need to redefine this command. Languages other than English may produce labels that are non-expandable, so we need to check for that otherwise it will create a “missing `\endcsname` inserted” error.

`\glsgetgrouptitle`

```
5518 \newcommand*{\glsgetgrouptitle}[1]{%
5519   \@gls@getgrouptitle{#1}{\@gls@grptitle}%
5520   \@gls@grptitle
5521 }
```

`\@gls@getgrouptitle` Gets the group title specified by the label (first argument) and stores in the second argument, which must be a control sequence.

```

5522 \newcommand*{\@gls@getgrouptitle}[2]{%
    Even if the argument appears to be a single letter, it won't be considered a single
    letter by \dtl@ifsingle if it's an active character.
5523 \dtl@ifsingle{#1}%
5524 {%
5525   \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5526 }%
5527 {%
5528   \ifboolexpr{test{\ifstrequal{#1}{glsymbols}}
5529               or test{\ifstrequal{#1}{glsnumbers}}}%
5530   {%
5531     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5532   }%
5533   {%
5534     \def#2{#1}%
5535   }%
5536 }%
5537 }

```

`@getothergrouptitle` Version for the no-indexing app option:

```

5538 \newcommand*{\@gls@noidx@getgrouptitle}[2]{%
5539   \DTLifint{#1}%
5540   {\edef#2{\char#1\relax}}%
5541   {%
5542     \ifcsundef{#1groupname}{\def#2{#1}}{\letcs#2{#1groupname}}%
5543   }%
5544 }

```

`\glsgetgrouplabel{<title>}`

This command does the reverse to the previous command. The argument is the group title, and it produces the group label. Note that if you redefine `\glsgetgrouptitle`, you will also need to redefine `\glsgetgrouplabel`.

`\glsgetgrouplabel`

```

5545 \newcommand*{\glsgetgrouplabel}[1]{%
5546 \ifthenelse{equal{#1}{\glsymbolsgroupname}}{glsymbols}{%
5547 \ifthenelse{equal{#1}{\glsnumbersgroupname}}{glsnumbers}{#1}}%

```

The command `\setentrycounter` sets the entry's associated counter (required by `\glshypernumber` etc.) `\glslink` and `\glsadd` encode the `\glossary` argument so that the relevant counter is set prior to the formatting command.

`\setentrycounter`

```

5548 \newcommand*{\setentrycounter}[2][ ]{%
5549   \def\@glo@counterprefix{#1}%
5550   \ifx\@glo@counterprefix\@empty

```

```

5551   \def\@glo@counterprefix{.}%
5552   \else
5553   \def\@glo@counterprefix{.#1.}%
5554   \fi
5555   \def\glseentrycounter{#2}%
5556 }

```

The current glossary style can be set using `\setglossarystyle{<style>}`.

`\setglossarystyle`

```

5557 \newcommand*\setglossarystyle[1]{%
5558   \ifcsundef{@glsstyle@#1}%
5559   {%
5560     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
5561   }%
5562   {%
5563     \csname @glsstyle@#1\endcsname
5564   }%
5565 }

```

`\glossarystyle`

```

5566 \newcommand*\glossarystyle[1]{%
5567   \ifcsundef{@glsstyle@#1}%
5568   {%
5569     \PackageError{glossaries}{Glossary style ‘#1’ undefined}{}%
5570   }%
5571   {%
5572     \GlossariesWarning
5573     {Deprecated command \string\glossarystyle.^^J
5574     I recommend you switch to \string\setglossarystyle\space unless
5575     you want to maintain backward compatibility}%
5576     \setglossentrycompatibility
5577     \csname @glsstyle@#1\endcsname

5578   \ifcsdef{@glscompstyle@#1}%
5579     {\setglossentrycompatibility\csuse{@glscompstyle@#1}}%
5580     {}%
5581   }%
5582 }

```

`\newglossarystyle` New glossary styles can be defined using:

```
\newglossarystyle{<name>}{<definition>}
```

The *<definition>* argument should redefine `theglossary`, `\glossaryheader`, `\glsgroupheading`, `\glossaryentryfield` and `\glsgroupskip` (see [subsection 1.18](#) for the definitions of predefined styles). Glossary styles should not redefine `\glossary preamble` and `\glossary postamble`, as the user should be able to switch between styles without affecting the pre- and postambles.

```

5583 \newcommand{\newglossarystyle}[2]{%
5584   \ifcsundef{@glsstyle@#1}%
5585   {%
5586     \expandafter\def\csname @glsstyle@#1\endcsname{#2}%
5587   }%
5588   {%
5589     \PackageError{glossaries}{Glossary style ‘#1’ is already defined}{}%
5590   }%
5591 }

```

`\renewglossarystyle` Code for this macro supplied by Marco Daniel.

```

5592 \newcommand{\renewglossarystyle}[2]{%
5593   \ifcsundef{@glsstyle@#1}%
5594   {%
5595     \PackageError{glossaries}{Glossary style ‘#1’ isn’t already defined}{}%
5596   }%
5597   {%
5598     \csdef{@glsstyle@#1}{#2}%
5599   }%
5600 }

```

Glossary entries are encoded so that the second argument to `\glossaryentryfield` is always specified as `\glsnamefont{<name>}`. This allows the user to change the font used to display the name term without having to redefine `\glossaryentryfield`. The default uses the surrounding font, so in the list type styles (which place the name in the optional argument to `\item`) the name will appear in bold.

`\glsnamefont`

```

5601 \newcommand*{\glsnamefont}[1]{#1}

```

Each glossary entry has an associated number list (usually page numbers) that indicate where in the document the entry has been used. The format for these number lists can be changed using the format key in commands like `\glslink`. The default format is given by `\glsnumber`. This takes a single argument which may be a single number, a number range or a number list. The number ranges are delimited with `\delimR`, the number lists are delimited with `\delimN`.

If the document doesn’t have hyperlinks, the numbers can be displayed just as they are, but if the document supports hyperlinks, the numbers should link to the relevant location. This means extracting the individual numbers from the list or ranges. The package does this with the `\hyperpage` command, but this is encoded for comma and dash delimiters and only for the page counter, but this code needs to be more general. So I have adapted the code used in the package.

`\glsnumber`

```

5602 \ifcsundef{hyperlink}%

```



```

5603 {%
5604 \def\glshypernumber#1{#1}%
5605 }%
5606 {%
5607 \def\glshypernumber#1{\@glshypernumber#1\nohyperpage{}}\@nil}
5608 }

```

`\@glshypernumber` This code was provided by Heiko Oberdiek to allow material to be attached to the location.

```

5609 \def\@glshypernumber#1\nohyperpage#2#3\@nil{%
5610 \ifx\#1\%
5611 \else
5612 \@delimR#1\delimR\delimR\%
5613 \fi
5614 \ifx\#2\%
5615 \else
5616 #2%
5617 \fi
5618 \ifx\#3\%
5619 \else
5620 \@glshypernumber#3\@nil
5621 \fi
5622 }

```

`\@delimR` displays a range of numbers for the counter whose name is given by `\@gls@counter` (which must be set prior to using `\glshypernumber`).

`\@delimR`

```

5623 \def\@delimR#1\delimR #2\delimR #3\%
5624 \ifx\#2\%
5625 \@delimN{#1}%
5626 \else
5627 \@gls@numberlink{#1}\delimR\@gls@numberlink{#2}%
5628 \fi}

```

`\@delimN` displays a list of individual numbers, instead of a range:

`\@delimN`

```

5629 \def\@delimN#1{\@delimN#1\delimN \delimN\}
5630 \def\@delimN#1\delimN #2\delimN#3\%
5631 \ifx\#3\%
5632 \@gls@numberlink{#1}%
5633 \else
5634 \@gls@numberlink{#1}\delimN\@gls@numberlink{#2}%
5635 \fi
5636 }

```

The following code is modified from hyperref's `\HyInd@pagelink` where the name of the counter being used is given by `\@gls@counter`.

```

5637 \def\@gls@numberlink#1{%
5638 \begingroup
5639 \toks@={}%
5640 \@gls@removespaces#1 \@nil
5641 \endgroup}

5642 \def\@gls@removespaces#1 #2\@nil{%
5643 \toks@=\expandafter{\the\toks@#1}%
5644 \ifx\#2\%
5645 \edef\x{\the\toks@}%
5646 \ifx\x\empty
5647 \else

5648 \hyperlink{\glstentrycounter\@glo@counterprefix\the\toks@}%
5649 {\the\toks@}%
5650 \fi
5651 \else
5652 \@gls@ReturnAfterFi{%
5653 \@gls@removespaces#2\@nil
5654 }%
5655 \fi
5656 }
5657 \long\def\@gls@ReturnAfterFi#1\fi{\fi#1}

```

The following commands will switch to the appropriate font, and create a hyperlink, if hyperlinks are supported. If hyperlinks are not supported, they will just display their argument in the appropriate font.

`\hyperrm`

```
5658 \newcommand*\hyperrm[1]{\textrm{\glshypernumber{#1}}}
```

`\hypersf`

```
5659 \newcommand*\hypersf[1]{\textsf{\glshypernumber{#1}}}
```

`\hypertt`

```
5660 \newcommand*\hypertt[1]{\texttt{\glshypernumber{#1}}}
```

`\hyperbf`

```
5661 \newcommand*\hyperbf[1]{\textbf{\glshypernumber{#1}}}
```

`\hypermd`

```
5662 \newcommand*\hypermd[1]{\textmd{\glshypernumber{#1}}}
```

`\hyperit`

```
5663 \newcommand*\hyperit[1]{\textit{\glshypernumber{#1}}}
```

`\hypersl`

```
5664 \newcommand*\hypersl[1]{\textsl{\glshypernumber{#1}}}
```

```
\hyperup
5665 \newcommand*{\hyperup}[1]{\textup{\glsnumber{#1}}}
```

```
\hypersc
5666 \newcommand*{\hypersc}[1]{\textsc{\glsnumber{#1}}}
```

```
\hyperemph
5667 \newcommand*{\hyperemph}[1]{\emph{\glsnumber{#1}}}
```

1.16 Acronyms

```
\oldacronym \oldacronym[⟨label⟩]{⟨abbrv⟩}{⟨long⟩}{⟨key-val list⟩}
```

This emulates the way the old package defined acronyms. It is equivalent to `\newacronym[⟨key-val list⟩]{⟨label⟩}{⟨abbrv⟩}{⟨long⟩}` and it additionally defines the command `\⟨label⟩` which is equivalent to `\gls{⟨label⟩}` (thus `⟨label⟩` must only contain alphabetical characters). If `⟨label⟩` is omitted, `⟨abbrv⟩` is used. This only emulates the syntax of the old package. The way the acronyms appear in the list of acronyms is determined by the definition of `\newacronym` and the glossary style.

Note that `\⟨label⟩` can't have an optional argument if the package is loaded. If hasn't been loaded then you can do `\⟨label⟩[⟨insert⟩]` but you can't do `\⟨label⟩[⟨key-val list⟩]`. For example if you define the acronym `svm`, then you can do `\svm['s]` but you can't do `\svm[format=textbf]`. If the package is loaded, `\svm['s]` will appear as `svm ['s]` which is unlikely to be the desired result. In this case, you will need to use `\gls` explicitly, e.g. `\gls{svm}['s]`. Note that it is up to the user to load if desired.

```
5668 \newcommand{\oldacronym}[4][\gls@label]{%
5669   \def\gls@label{#2}%
5670   \newacronym[#4]{#1}{#2}{#3}%
5671   \ifcsundef{xspace}%
5672     {%
5673       \expandafter\edef\csname#1\endcsname{%
5674         \noexpand\@ifstar{\noexpand\Gls{#1}}{\noexpand\gls{#1}}%
5675       }%
5676     }%
5677   {%
5678     \expandafter\edef\csname#1\endcsname{%
5679       \noexpand\@ifstar{\noexpand\Gls{#1}\noexpand\xspace}%
5680       \noexpand\gls{#1}\noexpand\xspace}%
5681     }%
5682   }%
5683 }
```

```
\newacronym[⟨key-val list⟩]{⟨label⟩}{⟨abbrev⟩}{⟨long⟩}
```

This is a quick way of defining acronyms, using `\newglossaryentry` with the appropriate values. It sets the glossary type to `\acronymtype` which will be acronym if the package option `acronym` has been used, otherwise it will be the default glossary. Since `\newacronym` merely calls `\newglossaryentry`, the acronym is treated like any other glossary entry.

If you prefer a different format, you can redefine `\newacronym` as required. The optional argument can be used to override any of the settings.

This is just a stub. It's redefined by commands like `\SetDefaultAcronymStyle`.

`\newacronym`

```
5684 \newcommand{\newacronym}[4] [] {}
```

Set up some convenient short cuts. These need to be changed if `\newacronym` is changed (or if the description key is changed).

`\acrpluralsuffix`

Plural suffix used by `\newacronym`. This just defaults to `\glspluralsuffix` but is changed to include `\textup` if the `smallcaps` option is used, so that the suffix doesn't appear in small caps as it doesn't look right. For example, `ABCs` looks as though the "s" is part of the acronym, but `ABCs` looks as though the "s" is a plural suffix. Since the entire text `abcs` is set in `\textsc`, `\textup` is need to cancel it out.

```
5685 \newcommand*{\acrpluralsuffix}{\glspluralsuffix}
```

If `garamondx` has been loaded, need to use `\textulc` instead of `\textup`.

`\glstextup`

```
5686 \newrobustcmd*{\glstextup}[1]{\ifdef\textulc{\textulc{#1}}{\textup{#1}}}
```

The following are defined for compatibility with version 2.07 and earlier.

`\glsshortkey`

```
5687 \newcommand*{\glsshortkey}{short}
```

`\glsshortpluralkey`

```
5688 \newcommand*{\glsshortpluralkey}{shortplural}
```

`\glslongkey`

```
5689 \newcommand*{\glslongkey}{long}
```

`\glslongpluralkey`

```
5690 \newcommand*{\glslongpluralkey}{longplural}
```

`\acrfull` Full form of the acronym.

```
5691 \newrobustcmd*{\acrfull}{\@gls@hyp@opt\ns@acrfull}
```

```
5692 \newcommand*\ns@acrfull[2] [] {%
```

```
5693 \new@ifnextchar[{\@acrfull{#1}{#2}}%
```

```
5694 \@acrfull{#1}{#2} []}%
```

```
5695 }
```

`\@acrfull` Low-level macro:

```
5696 \def\@acrfull#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5697 \acrfullfmt{#1}{#2}{#3}%  
5698 }
```

Using `\acrlinkfullformat` and `\acrfullformat` is now deprecated as it can cause complications with the first letter upper case variants, but the package needs to provide backward compatibility support.

`\acrfullfmt` No case change full format.

```
5699 \newcommand*\acrfullfmt}[3]{%  
5700 \acrlinkfullformat{\@acrlong}{\@acrshort}{#1}{#2}{#3}%  
5701 }
```

`\acrlinkfullformat` Format for full links like `\acrfull`. Syntax: `\acrlinkfullformat{<long cs>}{<short cs>}{<options>}{<label>}{<insert>}`

```
5702 \newcommand\acrlinkfullformat}[5]{%  
5703 \acrfullformat{#1{#3}{#4}[#5]}{#2{#3}{#4}[]}%  
5704 }
```

`\acrfullformat` Default full form is `<long>` (`<short>`).

```
5705 \newcommand\acrfullformat}[2]{#1\glsspace(#2)}
```

`\glsspace` Robust space to ensure it's written to the `.glsdefs` file.

```
5706 \newrobustcmd{\glsspace}{\space}
```

Default format for full acronym

`\Acrfull`

```
5707 \newrobustcmd*\Acrfull{\@gls@hyp@opt\ns@Acrfull}
```

```
5708 \newcommand*\ns@Acrfull[2][]{%
```

```
5709 \new@ifnextchar[{\@Acrfull{#1}{#2}}%
```

```
5710 \@Acrfull{#1}{#2}[]}%
```

```
5711 }
```

Low-level macro:

```
5712 \def\@Acrfull#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5713 \Acrfullfmt{#1}{#2}{#3}%  
5714 }
```

`\Acrfullfmt` First letter upper case full format.

```
5715 \newcommand*\Acrfullfmt}[3]{%
```

```
5716 \acrlinkfullformat{\@Acrlong}{\@acrshort}{#1}{#2}{#3}%
```

```
5717 }
```

`\ACRfull`

```
5718 \newrobustcmd*{\ACRfull}{\@gls@hyp@opt\ns@ACRfull}
```

```
5719 \newcommand*\ns@ACRfull[2] [] {%  
5720   \new@ifnextchar[{\@ACRfull{#1}{#2}}%  
5721     {\@ACRfull{#1}{#2} []}%  
5722 }
```

Low-level macro:

```
5723 \def\@ACRfull#1#2[#3] {%
```

Make it easier for acronym styles to change this:

```
5724   \ACRfullfmt{#1}{#2}{#3}%  
5725 }
```

`\ACRfullfmt` All upper case full format.

```
5726 \newcommand*\@ACRfullfmt[3] {%  
5727   \acrlinkfullformat{\@ACRlong}{\@ACRshort}{#1}{#2}{#3}%  
5728 }
```

Plural:

`\acrfullpl`

```
5729 \newrobustcmd*{\acrfullpl}{\@gls@hyp@opt\ns@acrfullpl}
```

```
5730 \newcommand*\ns@acrfullpl[2] [] {%  
5731   \new@ifnextchar[{\@acrfullpl{#1}{#2}}%  
5732     {\@acrfullpl{#1}{#2} []}%  
5733 }
```

Low-level macro:

```
5734 \def\@acrfullpl#1#2[#3] {%
```

Make it easier for acronym styles to change this:

```
5735   \acrfullplfmt{#1}{#2}{#3}%  
5736 }
```

`\acrfullplfmt` No case change plural full format.

```
5737 \newcommand*\@acrfullplfmt[3] {%  
5738   \acrlinkfullformat{\@acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%  
5739 }
```

`\Acrfullpl`

```
5740 \newrobustcmd*{\Acrfullpl}{\@gls@hyp@opt\ns@Acrfullpl}
```

```
5741 \newcommand*\ns@Acrfullpl[2] [] {%  
5742   \new@ifnextchar[{\@Acrfullpl{#1}{#2}}%  
5743     {\@Acrfullpl{#1}{#2} []}%  
5744 }
```

Low-level macro:

```
5745 \def\@Acrfullpl#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5746 \Acrfullplfmt{#1}{#2}{#3}%  
5747 }
```

`\Acrfullplfmt` First letter upper case plural full format.

```
5748 \newcommand*\Acrfullplfmt}[3]{%  
5749 \acrlinkfullformat{\@Acrlongpl}{\@acrshortpl}{#1}{#2}{#3}%  
5750 }
```

`\ACRfullpl`

```
5751 \newrobustcmd*\ACRfullpl{\@gls@hyp@opt\ns@ACRfullpl}  
  
5752 \newcommand*\ns@ACRfullpl[2][ ]{%  
5753 \new@ifnextchar[{\@ACRfullpl{#1}{#2}}%  
5754 {\@ACRfullpl{#1}{#2}[ ]}%  
5755 }
```

Low-level macro:

```
5756 \def\@ACRfullpl#1#2[#3]{%
```

Make it easier for acronym styles to change this:

```
5757 \ACRfullplfmt{#1}{#2}{#3}%  
5758 }
```

`\ACRfullplfmt` All upper case plural full format.

```
5759 \newcommand*\ACRfullplfmt}[3]{%  
5760 \acrlinkfullformat{\@ACRlongpl}{\@ACRshortpl}{#1}{#2}{#3}%  
5761 }
```

1.17 Predefined acronym styles

`\acronymfont` This is only used with the additional acronym styles:

```
5762 \newcommand{\acronymfont}[1]{#1}
```

`\firstacronymfont` This is only used with the additional acronym styles:

```
5763 \newcommand{\firstacronymfont}[1]{\acronymfont{#1}}
```

`\acrnameformat` The styles that allow an additional description use `\acrnameformat{<short>}{<long>}` to determine what information is displayed in the name.

```
5764 \newcommand*\acrnameformat}[2]{\acronymfont{#1}}
```

Define some tokens used by `\newacronym`:

`\glskeylisttok`

```
5765 \newtoks\glskeylisttok
```

`\glslabeltok`

5766 `\newtoks\glslabeltok`

`\glsshorttok`

5767 `\newtoks\glsshorttok`

`\glslongtok`

5768 `\newtoks\glslongtok`

`\newacronymhook` Provide a hook for `\newacronym`:

5769 `\newcommand*\newacronymhook{}`

`\SetGenericNewAcronym` New improved version of setting the acronym style.

5770 `\newcommand*\SetGenericNewAcronym{%`

Change the behaviour of `\Glsentryname` to workaround expansion issues that cause a problem for `\makefirstuc`

5771 `\let\@Gls@entryname\@Gls@acentryname`

Change the way acronyms are defined:

5772 `\renewcommand{\newacronym}[4][[]]{%`

5773 `\ifdefempty{\@glsacronymlists}%`

5774 `{%`

5775 `\def\@glo@type{\acronymtype}%`

5776 `\setkeys{glossentry}{##1}%`

5777 `\DeclareAcronymList{\@glo@type}%`

5778 `}%`

5779 `{}%`

5780 `\glskeylisttok{##1}%`

5781 `\glslabeltok{##2}%`

5782 `\glsshorttok{##3}%`

5783 `\glslongtok{##4}%`

5784 `\newacronymhook`

5785 `\protected@edef\@do@newglossaryentry{%`

5786 `\noexpand\newglossaryentry{\the\glslabeltok}%`

5787 `{%`

5788 `type=\acronymtype,%`

5789 `name={\expandonce{\acronymentry{##2}}},%`

5790 `sort={\acronymssort{\the\glsshorttok}{\the\glslongtok}},%`

5791 `text={\the\glsshorttok},%`

5792 `short={\the\glsshorttok},%`

5793 `shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%`

5794 `long={\the\glslongtok},%`

5795 `longplural={\the\glslongtok\noexpand\acrpluralsuffix},%`

5796 `\GenericAcronymFields,%`

5797 `\the\glskeylisttok`

5798 `}%`

5799 `{}%`

5800 `\@do@newglossaryentry`

5801 `}%`

Make sure that `\acrfull` etc reflects the new style:

```

5802 \renewcommand*\acrfullfmt}[3]{%
5803   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
5804 \renewcommand*\acrfullfmt}[3]{%
5805   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
5806 \renewcommand*\ACRfullfmt}[3]{%
5807   \glslink[##1]{##2}{%
5808     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}}}%
5809 \renewcommand*\acrfullplfmt}[3]{%
5810   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
5811 \renewcommand*\Acrfullplfmt}[3]{%
5812   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
5813 \renewcommand*\ACRfullplfmt}[3]{%
5814   \glslink[##1]{##2}{%
5815     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}}}%

```

Make sure that `\glentryfull` etc reflects the new style:

```

5816 \renewcommand*\glentryfull}[1]{\genacrfullformat{##1}{}}}%
5817 \renewcommand*\Glsentryfull}[1]{\Genacrfullformat{##1}{}}}%
5818 \renewcommand*\glentryfullpl}[1]{\genplacrfullformat{##1}{}}}%
5819 \renewcommand*\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}}%
5820 }

```

`\GenericAcronymFields` Fields used by `\SetGenericNewAcronym` that can be changed by the acronym style.

```
5821 \newcommand*\GenericAcronymFields}{description={\the\glslongtok}}
```

`\acronymentry` `\acronymentry{<label>}`

Display style for the name field in the list of acronyms.

```
5822 \newcommand*\acronymentry}[1]{\acronymfont{\glentryshort{#1}}}
```

`\acronymsort` `\acronymsort{<short>}{<long>}`

Default sort format for acronyms.

```
5823 \newcommand*\acronymsort}[2]{#1}
```

`\setacronymstyle` `\setacronymstyle{<style name>}`

```

5824 \newcommand*\setacronymstyle}[1]{%
5825   \ifcsundef{@glsacr@dispstyle@#1}%
5826   {%
5827     \PackageError{glossaries}{Undefined acronym style ‘#1’}{%
5828     }%

```

```

5829  {%
5830    \ifdefempty{\@glsacronymlists}%
5831    {%
5832      \DeclareAcronymList{\acronymtype}%
5833    }%
5834  }%
5835  \SetGenericNewAcronym
5836  \GlsUseAcrStyleDefs{#1}%
5837  \@for\@gls@type:=\@glsacronymlists\do{%
5838    \defglsentryfmt[\@gls@type]{\GlsUseAcrEntryDisplayStyle{#1}}%
5839  }%
5840 }%
5841 }

```

`\newacronymstyle` `\newacronymstyle{<style name>}{<entry format definition>}{<display definitions>}`

Defines a new acronym style called *<style name>*.

```

5842 \newcommand*\newacronymstyle}[3]{%
5843   \ifcsdef{@glsacr@dispstyle@#1}%
5844   {%
5845     \PackageError{glossaries}{Acronym style ‘#1’ already exists}{}%
5846   }%
5847   {%
5848     \csdef{@glsacr@dispstyle@#1}{#2}%
5849     \csdef{@glsacr@styledefs@#1}{#3}%
5850   }%
5851 }

```

`\renewacronymstyle` Redefines the given acronym style.

```

5852 \newcommand*\renewacronymstyle}[3]{%
5853   \ifcsdef{@glsacr@dispstyle@#1}%
5854   {%
5855     \csdef{@glsacr@dispstyle@#1}{#2}%
5856     \csdef{@glsacr@styledefs@#1}{#3}%
5857   }%
5858   {%
5859     \PackageError{glossaries}{Acronym style ‘#1’ doesn’t exist}{}%
5860   }%
5861 }

```

`\useAcrEntryDisplayStyle`

```

5862 \newcommand*\GlsUseAcrEntryDisplayStyle}[1]{\csuse{@glsacr@dispstyle@#1}}

```

`\GlsUseAcrStyleDefs`

```

5863 \newcommand*\GlsUseAcrStyleDefs}[1]{\csuse{@glsacr@styledefs@#1}}

```

Predefined acronym styles:

long-short *<long>* (*<short>*) acronym style.

```
5864 \newacronymstyle{long-short}%
5865 {%
    Check for long form in case this is a mixed glossary.
5866 \ifglshaslong{\glslabel}{\glsngenacfmt}{\glsngenentryfmt}%
5867 }%
5868 {%
5869 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5870 \renewcommand*{\genacrfullformat}[2]{%
5871 \glsentrylong{##1}##2\space
5872 (\protect\firstacronymfont{\glsentryshort{##1}})%
5873 }%
5874 \renewcommand*{\Genacrfullformat}[2]{%
5875 \Glsentrylong{##1}##2\space
5876 (\protect\firstacronymfont{\glsentryshort{##1}})%
5877 }%
5878 \renewcommand*{\genplacrfullformat}[2]{%
5879 \glsentrylongpl{##1}##2\space
5880 (\protect\firstacronymfont{\glsentryshortpl{##1}})%
5881 }%
5882 \renewcommand*{\Genplacrfullformat}[2]{%
5883 \Glsentrylongpl{##1}##2\space
5884 (\protect\firstacronymfont{\glsentryshortpl{##1}})%
5885 }%
5886 \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
5887 \renewcommand*{\acronymsort}[2]{##1}%
5888 \renewcommand*{\acronymfont}[1]{##1}%
5889 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
5890 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
5891 }
```

short-long *<short>* (*<long>*) acronym style.

```
5892 \newacronymstyle{short-long}%
5893 {%
    Check for long form in case this is a mixed glossary.
5894 \ifglshaslong{\glslabel}{\glsngenacfmt}{\glsngenentryfmt}%
5895 }%
5896 {%
5897 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
5898 \renewcommand*{\genacrfullformat}[2]{%
5899 \protect\firstacronymfont{\glsentryshort{##1}}##2\space
5900 (\glsentrylong{##1})%
5901 }%
5902 \renewcommand*{\Genacrfullformat}[2]{%
5903 \protect\firstacronymfont{\Glsentryshort{##1}}##2\space
5904 (\glsentrylong{##1})%
5905 }%
5906 \renewcommand*{\genplacrfullformat}[2]{%

```

```

5907 \protect\firstacronymfont{\glentryshortpl{##1}}##2\space
5908 (\glentrylongpl{##1})%
5909 }%
5910 \renewcommand*\Genplacrfullformat}[2]{%
5911 \protect\firstacronymfont{\glentryshortpl{##1}}##2\space
5912 (\glentrylongpl{##1})%
5913 }%

5914 \renewcommand*\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}%
5915 \renewcommand*\acronymsort}[2]{##1}%
5916 \renewcommand*\acronymfont}[1]{##1}%
5917 \renewcommand*\firstacronymfont}[1]{\acronymfont{##1}}%
5918 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
5919 }

```

long-sc-short *<long>* (\textsc{<short>}) acronym style.

```

5920 \newacronymstyle{long-sc-short}%
5921 {%
5922 \GlsUseAcrEntryDispStyle{long-short}%
5923 }%
5924 {%
5925 \GlsUseAcrStyleDefs{long-short}%
5926 \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5927 \renewcommand*\acrpluralsuffix{\glstextup{\glspluralsuffix}}%
5928 }

```

long-sm-short *<long>* (\textsmaller{<short>}) acronym style.

```

5929 \newacronymstyle{long-sm-short}%
5930 {%
5931 \GlsUseAcrEntryDispStyle{long-short}%
5932 }%
5933 {%
5934 \GlsUseAcrStyleDefs{long-short}%
5935 \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5936 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
5937 }

```

sc-short-long *<short>* (\textsc{<long>}) acronym style.

```

5938 \newacronymstyle{sc-short-long}%
5939 {%
5940 \GlsUseAcrEntryDispStyle{short-long}%
5941 }%
5942 {%
5943 \GlsUseAcrStyleDefs{short-long}%
5944 \renewcommand{\acronymfont}[1]{\textsc{##1}}%
5945 \renewcommand*\acrpluralsuffix{\glstextup{\glspluralsuffix}}%
5946 }

```

sm-short-long *<short>* (\textsmaller{<long>}) acronym style.

```

5947 \newacronymstyle{sm-short-long}%
5948 {%
5949   \GlsUseAcrEntryDispStyle{short-long}%
5950 }%
5951 {%
5952   \GlsUseAcrStyleDefs{short-long}%
5953   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
5954   \renewcommand*\{acrpluralsuffix}{\glspluralsuffix}%
5955 }

```

long-short-desc *⟨long⟩* (*⟨short⟩*) acronym style that has an accompanying description (which the user needs to supply).

```

5956 \newacronymstyle{long-short-desc}%
5957 {%
5958   \GlsUseAcrEntryDispStyle{long-short}%
5959 }%
5960 {%
5961   \GlsUseAcrStyleDefs{long-short}%
5962   \renewcommand*\{GenericAcronymFields}{}%
5963   \renewcommand*\{acronymsort}[2]{##2}%
5964   \renewcommand*\{acronymentry}[1]{%
5965     \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
5966 }

```

long-sc-short-desc *⟨long⟩* (*\textsc{⟨short⟩}*) acronym style that has an accompanying description (which the user needs to supply).

```

5967 \newacronymstyle{long-sc-short-desc}%
5968 {%
5969   \GlsUseAcrEntryDispStyle{long-sc-short}%
5970 }%
5971 {%
5972   \GlsUseAcrStyleDefs{long-sc-short}%
5973   \renewcommand*\{GenericAcronymFields}{}%
5974   \renewcommand*\{acronymsort}[2]{##2}%
5975   \renewcommand*\{acronymentry}[1]{%
5976     \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
5977 }

```

long-sm-short-desc *⟨long⟩* (*\textsmaller{⟨short⟩}*) acronym style that has an accompanying description (which the user needs to supply).

```

5978 \newacronymstyle{long-sm-short-desc}%
5979 {%
5980   \GlsUseAcrEntryDispStyle{long-sm-short}%
5981 }%
5982 {%
5983   \GlsUseAcrStyleDefs{long-sm-short}%
5984   \renewcommand*\{GenericAcronymFields}{}%
5985   \renewcommand*\{acronymsort}[2]{##2}%
5986   \renewcommand*\{acronymentry}[1]{%

```

```
5987 \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
5988 }
```

short-long-desc <short> ({<long>}) acronym style that has an accompanying description (which the user needs to supply).

```
5989 \newacronymstyle{short-long-desc}%
5990 {%
5991 \GlsUseAcrEntryDispStyle{short-long}%
5992 }%
5993 {%
5994 \GlsUseAcrStyleDefs{short-long}%
5995 \renewcommand*\GenericAcronymFields{}%
5996 \renewcommand*\acronymsort}[2]{##2}%
5997 \renewcommand*\acronymentry}[1]{%
5998 \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
5999 }
```

sc-short-long-desc <long> (\textsc{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```
6000 \newacronymstyle{sc-short-long-desc}%
6001 {%
6002 \GlsUseAcrEntryDispStyle{sc-short-long}%
6003 }%
6004 {%
6005 \GlsUseAcrStyleDefs{sc-short-long}%
6006 \renewcommand*\GenericAcronymFields{}%
6007 \renewcommand*\acronymsort}[2]{##2}%
6008 \renewcommand*\acronymentry}[1]{%
6009 \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
6010 }
```

sm-short-long-desc <long> (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```
6011 \newacronymstyle{sm-short-long-desc}%
6012 {%
6013 \GlsUseAcrEntryDispStyle{sm-short-long}%
6014 }%
6015 {%
6016 \GlsUseAcrStyleDefs{sm-short-long}%
6017 \renewcommand*\GenericAcronymFields{}%
6018 \renewcommand*\acronymsort}[2]{##2}%
6019 \renewcommand*\acronymentry}[1]{%
6020 \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
6021 }
```

dua <long> only acronym style.

```
6022 \newacronymstyle{dua}%
6023 }
```

Check for long form in case this is a mixed glossary.

```
6024 \ifdefempty\glscustomtext
6025 {%
6026   \ifglshaslong{\glslabel}%
6027   {%
6028     \glsifplural
6029     {%
```

Plural form:

```
6030     \glscapscase
6031     {%
```

Plural form, don't adjust case:

```
6032     \glstentrylongpl{\glslabel}\glsinsert
6033     }%
6034     {%
```

Plural form, make first letter upper case:

```
6035     \Glstentrylongpl{\glslabel}\glsinsert
6036     }%
6037     {%
```

Plural form, all caps:

```
6038     \mfirstucMakeUppercase
6039     {\glstentrylongpl{\glslabel}\glsinsert}%
6040     }%
6041     }%
6042     {%
```

Singular form

```
6043     \glscapscase
6044     {%
```

Singular form, don't adjust case:

```
6045     \glstentrylong{\glslabel}\glsinsert
6046     }%
6047     {%
```

Subsequent singular form, make first letter upper case:

```
6048     \Glstentrylong{\glslabel}\glsinsert
6049     }%
6050     {%
```

Subsequent singular form, all caps:

```
6051     \mfirstucMakeUppercase
6052     {\glstentrylong{\glslabel}\glsinsert}%
6053     }%
6054     }%
6055     }%
6056     {%
```

Not an acronym:

```
6057     \glsgenentryfmt
6058     }%
6059     }%
6060     {\glscustomtext\glsinsert}%
6061 }%
6062 {%
6063     \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%

6064     \renewcommand*{\acrfullfmt}[3]{%
6065         \glslink[##1]{##2}{\glsentrylong{##2}##3\space
6066             (\acronymfont{\glsentryshort{##2}})}}%
6067     \renewcommand*{\Acrfullfmt}[3]{%
6068         \glslink[##1]{##2}{\Glsentrylong{##2}##3\space
6069             (\acronymfont{\glsentryshort{##2}})}}%
6070     \renewcommand*{\ACRfullfmt}[3]{%
6071         \glslink[##1]{##2}{%
6072             \mfirstucMakeUppercase{\glsentrylong{##2}##3\space
6073                 (\acronymfont{\glsentryshort{##2}})}}}%

6074     \renewcommand*{\acrfullplfmt}[3]{%
6075         \glslink[##1]{##2}{\glsentrylongpl{##2}##3\space
6076             (\acronymfont{\glsentryshorttpl{##2}})}}%

6077     \renewcommand*{\Acrfullplfmt}[3]{%
6078         \glslink[##1]{##2}{\Glsentrylongpl{##2}##3\space
6079             (\acronymfont{\glsentryshorttpl{##2}})}}%
6080     \renewcommand*{\ACRfullplfmt}[3]{%
6081         \glslink[##1]{##2}{%
6082             \mfirstucMakeUppercase{\glsentrylongpl{##2}##3\space
6083                 (\acronymfont{\glsentryshorttpl{##2}})}}}%
6084     \renewcommand*{\glsentryfull}[1]{%
6085         \glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
6086     }%
6087     \renewcommand*{\Glsentryfull}[1]{%
6088         \Glsentrylong{##1}\space(\acronymfont{\glsentryshort{##1}})%
6089     }%
6090     \renewcommand*{\glsentryfullpl}[1]{%
6091         \glsentrylongpl{##1}\space(\acronymfont{\glsentryshorttpl{##1}})%
6092     }%
6093     \renewcommand*{\Glsentryfullpl}[1]{%
6094         \Glsentrylongpl{##1}\space(\acronymfont{\glsentryshorttpl{##1}})%
6095     }%
6096     \renewcommand*{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}%
6097     \renewcommand*{\acronymsort}[2]{##1}%
6098     \renewcommand*{\acronymfont}[1]{##1}%
6099     \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
6100 }
```


dua-desc *<long>* only acronym style with user-supplied description.

```
6101 \newacronymstyle{dua-desc}%
6102 {%
6103   \GlsUseAcrEntryDispStyle{dua}%
6104 }%
6105 {%
6106   \GlsUseAcrStyleDefs{dua}%
6107   \renewcommand*{\GenericAcronymFields}{}%

6108   \renewcommand*{\acronymentry}[1]{\acronymfont{\glentrylong{##1}}}%
6109   \renewcommand*{\acronymsort}[2]{##2}%
6110 }%
```

footnote *<short>*\footnote{*<long>*} acronym style.

```
6111 \newacronymstyle{footnote}%
6112 {%

  Check for long form in case this is a mixed glossary.

6113   \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
6114 }%
6115 {%
6116   \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
```

Need to ensure hyperlinks are switched off on first use:

```
6117   \glshyperfirstfalse
6118   \renewcommand*{\genacrfullformat}[2]{%
6119     \protect\firstacronymfont{\glentryshort{##1}}##2%
6120     \protect\footnote{\glentrylong{##1}}%
6121   }%
6122   \renewcommand*{\Genacrfullformat}[2]{%
6123     \firstacronymfont{\Glsentryshort{##1}}##2%
6124     \protect\footnote{\glentrylong{##1}}%
6125   }%
6126   \renewcommand*{\genplacrfullformat}[2]{%
6127     \protect\firstacronymfont{\glentryshortpl{##1}}##2%
6128     \protect\footnote{\glentrylongpl{##1}}%
6129   }%
6130   \renewcommand*{\Genplacrfullformat}[2]{%
6131     \protect\firstacronymfont{\Glsentryshortpl{##1}}##2%
6132     \protect\footnote{\glentrylongpl{##1}}%
6133   }%
6134   \renewcommand*{\acronymentry}[1]{\acronymfont{\glentryshort{##1}}}%
6135   \renewcommand*{\acronymsort}[2]{##1}%
6136   \renewcommand*{\acronymfont}[1]{##1}%
6137   \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
```

Don't use footnotes for \acrfull:

```
6138   \renewcommand*{\acrfullfmt}[3]{%
6139     \glslink[##1]{##2}{\acronymfont{\glentryshort{##2}}##3\space
6140       (\glentrylong{##2})}%
```

```

6141 \renewcommand*\Acrfullfmt}[3]{%
6142   \glslink[##1]{##2}{\acronymfont{\Glsentryshort{##2}}##3\space
6143   (\glsentrylong{##2})}%
6144 \renewcommand*\ACRfullfmt}[3]{%
6145   \glslink[##1]{##2}{%
6146   \mfirstucMakeUppercase{\acronymfont{\glsentryshort{##2}}##3\space
6147   (\glsentrylong{##2})}}}%
6148 \renewcommand*\acrfullplfmt}[3]{%
6149   \glslink[##1]{##2}{\acronymfont{\glsentryshortpl{##2}}##3\space
6150   (\glsentrylongpl{##2})}%
6151 \renewcommand*\Acrfullplfmt}[3]{%
6152   \glslink[##1]{##2}{\acronymfont{\Glsentryshortpl{##2}}##3\space
6153   (\glsentrylongpl{##2})}%
6154 \renewcommand*\ACRfullplfmt}[3]{%
6155   \glslink[##1]{##2}{%
6156   \mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{##2}}##3\space
6157   (\glsentrylongpl{##2})}}}%

```

Similarly for \glsentryfull etc:

```

6158 \renewcommand*\glsentryfull}[1]{%
6159   \acronymfont{\glsentryshort{##1}}\space(\glsentrylong{##1})}%
6160 \renewcommand*\Glsentryfull}[1]{%
6161   \acronymfont{\Glsentryshort{##1}}\space(\glsentrylong{##1})}%
6162 \renewcommand*\glsentryfullpl}[1]{%
6163   \acronymfont{\glsentryshortpl{##1}}\space(\glsentrylongpl{##1})}%
6164 \renewcommand*\Glsentryfullpl}[1]{%
6165   \acronymfont{\Glsentryshortpl{##1}}\space(\glsentrylongpl{##1})}%
6166 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

6167 \newacronymstyle{footnote-sc}%
6168 {%
6169   \GlsUseAcrEntryDispStyle{footnote}%
6170 }%
6171 {%
6172   \GlsUseAcrStyleDefs{footnote}%
6173   \renewcommand{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
6174   \renewcommand{\acronymfont}[1]{\textsc{##1}}%
6175   \renewcommand*\acrpluralsuffix{\glsstextup{\glspluralsuffix}}%
6176 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```

6177 \newacronymstyle{footnote-sm}%
6178 {%
6179   \GlsUseAcrEntryDispStyle{footnote}%
6180 }%
6181 {%
6182   \GlsUseAcrStyleDefs{footnote}%
6183   \renewcommand{\acronymentry}[1]{\acronymfont{\glsentryshort{##1}}}
6184   \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%

```

```
6185 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
6186 }%
```

footnote-desc <short>\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```
6187 \newacronymstyle{footnote-desc}%
6188 {%
6189 \GlsUseAcrEntryDispStyle{footnote}%
6190 }%
6191 {%
6192 \GlsUseAcrStyleDefs{footnote}%
6193 \renewcommand*{\GenericAcronymFields}{}%
6194 \renewcommand*{\acronymsort}[2]{##2}%
6195 \renewcommand*{\acronymentry}[1]{%
6196 \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
6197 }
```

footnote-sc-desc \textsc{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```
6198 \newacronymstyle{footnote-sc-desc}%
6199 {%
6200 \GlsUseAcrEntryDispStyle{footnote-sc}%
6201 }%
6202 {%
6203 \GlsUseAcrStyleDefs{footnote-sc}%
6204 \renewcommand*{\GenericAcronymFields}{}%
6205 \renewcommand*{\acronymsort}[2]{##2}%
6206 \renewcommand*{\acronymentry}[1]{%
6207 \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
6208 }
```

footnote-sm-desc \textsmaller{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```
6209 \newacronymstyle{footnote-sm-desc}%
6210 {%
6211 \GlsUseAcrEntryDispStyle{footnote-sm}%
6212 }%
6213 {%
6214 \GlsUseAcrStyleDefs{footnote-sm}%
6215 \renewcommand*{\GenericAcronymFields}{}%
6216 \renewcommand*{\acronymsort}[2]{##2}%
6217 \renewcommand*{\acronymentry}[1]{%
6218 \glsentrylong{##1}\space (\acronymfont{\glsentryshort{##1}})}%
6219 }
```

fineAcronymSynonyms

```
6220 \newcommand*{\DefineAcronymSynonyms}{%
```

Short form

`\acs`

6221 `\let\acs\acrshort`

First letter uppercase short form

`\Acs`

6222 `\let\Acs\Acrshort`

Plural short form

`\acsp`

6223 `\let\acsp\acrshortpl`

First letter uppercase plural short form

`\Acsp`

6224 `\let\Acsp\Acrshortpl`

Long form

`\acl`

6225 `\let\acl\acrlong`

Plural long form

`\aclp`

6226 `\let\aclp\acrlongpl`

First letter upper case long form

`\Acl`

6227 `\let\Acl\Acrlong`

First letter upper case plural long form

`\Aclp`

6228 `\let\Aclp\Acrlongpl`

Full form

`\acf`

6229 `\let\acf\acrfull`

Plural full form

`\acfp`

6230 `\let\acfp\acrfullpl`

First letter upper case full form

`\Acf`

6231 `\let\Acf\Acrfull`

First letter upper case plural full form

`\Acfp`

```
6232 \let\Acfp\Acrfullpl
```

Standard form

`\ac`

```
6233 \let\ac\gls
```

First upper case standard form

`\Ac`

```
6234 \let\Ac\Gls
```

Standard plural form

`\acp`

```
6235 \let\acp\glspl
```

Standard first letter upper case plural form

`\Acp`

```
6236 \let\Acp\Glspl
```

```
6237 }
```

Define synonyms if required

```
6238 \ifglsacrshortcuts
```

```
6239 \DefineAcronymSynonyms
```

```
6240 \fi
```

These commands for setting the style are now deprecated but are kept for backward compatibility.

`AcronymDisplayStyle` Sets the default acronym display style for given glossary.

```
6241 \newcommand*\SetDefaultAcronymDisplayStyle}[1]{%
```

```
6242 \defglsentryfmt[#1]{\glsgenentryfmt}%
```

```
6243 }
```

`defaultNewAcronymDef` Sets up the acronym definition for the default style. The information is provided by the tokens `\glslabeltok`, `\glsshorttok`, `\gslongtok` and `\glskeylisttok`.

```
6244 \newcommand*\DefaultNewAcronymDef){%
```

```
6245 \edef\do@newglossaryentry{%
```

```
6246 \noexpand\newglossaryentry{\the\glslabeltok}%
```

```
6247 {%
```

```
6248 type=\acronymtype,%
```

```
6249 name={\the\glsshorttok},%
```

```
6250 sort={\the\glsshorttok},%
```

```
6251 text={\the\glsshorttok},%
```

```

6252     first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
6253     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6254     firstplural={\acrfullformat{\noexpand\expandonce\noexpand\@glo@longpl}%
6255                 {\noexpand\expandonce\noexpand\@glo@shortpl}},%
6256     short={\the\glsshorttok},%
6257     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6258     long={\the\glslongtok},%
6259     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6260     description={\the\glslongtok},%
6261     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%

```

Remaining options specified by the user:

```

6262     \the\glskeylisttok
6263 }%
6264 }%
6265 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6266 \let\@org@gls@assign@plural\gls@assign@plural
6267 \let\@org@gls@assign@descplural\gls@assign@descplural
6268 \def\gls@assign@firstpl##1##2{%
6269     \@@gls@expand@field{##1}{firstpl}{##2}%
6270 }%
6271 \def\gls@assign@plural##1##2{%
6272     \@@gls@expand@field{##1}{plural}{##2}%
6273 }%
6274 \def\gls@assign@descplural##1##2{%
6275     \@@gls@expand@field{##1}{descplural}{##2}%
6276 }%
6277 \do@newglossaryentry
6278 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6279 \let\gls@assign@plural\@org@gls@assign@plural
6280 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6281 }

```

DefaultAcronymStyle Set up the default acronym style:

```

6282 \newcommand*\SetDefaultAcronymStyle}{%

```

Set the display style:

```

6283 \@for\@gls@type:=\@glsacronymlists\do{%
6284     \SetDefaultAcronymDisplayStyle{\@gls@type}%
6285 }%

```

Set up the definition of `\newacronym`:

```

6286 \renewcommand{\newacronym}[4][[]]{%

```

If user is just using the main glossary and hasn't identified it as a list of acronyms, then update. (This is done to ensure backwards compatibility with versions prior to 2.04).

```

6287     \ifx\@glsacronymlists\@empty
6288         \def\@glo@type{\acronymtype}%
6289         \setkeys{glossentry}{##1}%
6290         \DeclareAcronymList{\@glo@type}%

```

```

6291     \SetDefaultAcronymDisplayStyle{\@glo@type}%
6292     \fi
6293     \glskeylisttok{##1}%
6294     \glslabeltok{##2}%
6295     \glsshorttok{##3}%
6296     \gslongtok{##4}%
6297     \newacronymhook
6298     \DefaultNewAcronymDef
6299 }%
6300 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
6301 }

```

`\acrfootnote` Used by the footnote acronym styles.

```
6302 \newcommand*{\acrfootnote}[3]{\acrlinkfootnote{#1}{#2}{#3}}
```

`\acrlinkfootnote`

```

6303 \newcommand*{\acrlinkfootnote}[3]{%
6304   \footnote{\glslink[#1]{#2}{#3}}%
6305 }

```

`\acrnoflinkfootnote`

```

6306 \newcommand*{\acrnoflinkfootnote}[3]{%
6307   \footnote{#3}%
6308 }

```

`AcronymDisplayStyle` Sets the acronym display style for given glossary for the description and footnote combination.

```

6309 \newcommand*{\SetDescriptionFootnoteAcronymDisplayStyle}[1]{%
6310   \defglsentryfmt[#1]{%

```

```

6311     \ifdefempty\glscustomtext
6312     {%
6313       \ifglsused{\glslabel}%
6314       {%
6315         \acronymfont{\glsentryfmt}%
6316       }%
6317     {%
6318       \firstacronymfont{\glsentryfmt}%
6319       \ifglsymbol{\glslabel}%
6320       {%
6321         \expandafter\protect\expandafter\acrfootnote\expandafter
6322         {\@gls@link@opts}{\@gls@link@label}%
6323       }%
6324       \glsifplural
6325       {\glsentrysymbolplural{\glslabel}}%
6326       {\glsentrysymbol{\glslabel}}%
6327     }%
6328   }%
6329 }%

```

```

6330 }%
6331 {\glscustomtext\glsinsert}%
6332 }%
6333 }

```

otnoteNewAcronymDef

```

6334 \newcommand*{\DescriptionFootnoteNewAcronymDef}{%
6335 \edef\@do@newglossaryentry{%
6336 \noexpand\newglossaryentry{\the\glslabeltok}%
6337 {%
6338 type=\acronymtype,%
6339 name={\noexpand\acronymfont{\the\glsshorttok}},%
6340 sort={\the\glsshorttok},%
6341 first={\the\glsshorttok},%
6342 firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6343 text={\the\glsshorttok},%
6344 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6345 short={\the\glsshorttok},%
6346 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6347 long={\the\glslongtok},%
6348 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6349 symbol={\the\glslongtok},%
6350 symbolplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6351 \the\glskeylisttok
6352 }%
6353 }%
6354 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6355 \let\@org@gls@assign@plural\gls@assign@plural
6356 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6357 \def\gls@assign@firstpl##1##2{%
6358 \@@gls@expand@field{##1}{firstpl}{##2}%
6359 }%
6360 \def\gls@assign@plural##1##2{%
6361 \@@gls@expand@field{##1}{plural}{##2}%
6362 }%
6363 \def\gls@assign@symbolplural##1##2{%
6364 \@@gls@expand@field{##1}{symbolplural}{##2}%
6365 }%
6366 \@do@newglossaryentry
6367 \let\gls@assign@plural\@org@gls@assign@plural
6368 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6369 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6370 }

```

ootnoteAcronymStyle

If a description and footnote are both required, store the long form in the symbol key. Store the short form in text key. Note that since the long form is stored in the symbol key, if you want the long form to appear in the list of acronyms, you need to use a glossary style that displays the symbol key.

```

6371 \newcommand*{\SetDescriptionFootnoteAcronymStyle}{%

```



```

6372 \renewcommand{\newacronym}[4][\]{%
6373   \ifx\@glsacronymlists\@empty
6374     \def\@glo@type{\acronymtype}%
6375     \setkeys{glossentry}{##1}%
6376     \DeclareAcronymList{\@glo@type}%
6377     \SetDescriptionFootnoteAcronymDisplayStyle{\@glo@type}%
6378     \fi
6379     \glskeylisttok{##1}%
6380     \glslabeltok{##2}%
6381     \glsshorttok{##3}%
6382     \glslongtok{##4}%
6383     \newacronymhook
6384     \DescriptionFootnoteNewAcronymDef
6385   }%

```

If footnote package option is specified, set the first use to append the long form (stored in symbol) as a footnote.

```

6386 \@for\@gls@type:=\@glsacronymlists\do{%
6387   \SetDescriptionFootnoteAcronymDisplayStyle{\@gls@type}%
6388 }%

```

Redefine `\acronymfont` if small caps required. The plural suffix is set in an upright font so that it remains in normal lower case, otherwise it looks as though it's part of the acronym.

```

6389 \ifglsacrsmallcaps
6390   \renewcommand*\acronymfont}[1]{\textsc{##1}}%
6391   \renewcommand*\acrpluralsuffix}{%
6392     \glstextup{\glspluralsuffix}}%
6393 \else
6394   \ifglsacrsmaller
6395     \renewcommand*\acronymfont}[1]{\textsmaller{##1}}%
6396   \fi
6397 \fi

```

Check for package option clash

```

6398 \ifglsacrdua
6399   \PackageError{glossaries}{Option clash: 'footnote' and 'dua'
6400     can't both be set}{}%
6401   \fi
6402 }%

```

`AcronymDisplayStyle` Sets the acronym display style for given glossary with description and dua combination.

```

6403 \newcommand*\SetDescriptionDUAAcronymDisplayStyle}[1]{%
6404   \def\glsentryfmt[#1]{\glsentryfmt}%
6405 }

```

`ionDUANewAcronymDef`

```

6406 \newcommand*\DescriptionDUANewAcronymDef}{%

```

```

6407 \edef\@do@newglossaryentry{%
6408   \noexpand\newglossaryentry{\the\glslabeltok}%
6409   {%
6410     type=\acronymtype,%
6411     name={\the\glslongtok},%
6412     sort={\the\glslongtok},
6413     text={\the\glslongtok},%
6414     first={\the\glslongtok},%
6415     plural={\noexpand\expandonce\noexpand\@glo@longpl},%
6416     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6417     short={\the\glsshorttok},%
6418     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6419     long={\the\glslongtok},%
6420     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6421     symbol={\the\glsshorttok},%
6422     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6423     \the\glskeylisttok
6424   }%
6425 }%
6426 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6427 \let\@org@gls@assign@plural\gls@assign@plural
6428 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6429 \def\gls@assign@firstpl##1##2{%
6430   \@@gls@expand@field{##1}{firstpl}{##2}%
6431 }%
6432 \def\gls@assign@plural##1##2{%
6433   \@@gls@expand@field{##1}{plural}{##2}%
6434 }%
6435 \def\gls@assign@symbolplural##1##2{%
6436   \@@gls@expand@field{##1}{symbolplural}{##2}%
6437 }%
6438 \@do@newglossaryentry
6439 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6440 \let\gls@assign@plural\@org@gls@assign@plural
6441 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6442 }

```

tionDUAAcronymStyle Description, don't use acronym and no footnote. Note that the short form is stored in the symbol key, so if the short form needs to be displayed in the glossary, use a style the displays the symbol.

```

6443 \newcommand*\SetDescriptionDUAAcronymStyle{%
6444   \ifglsacrsmallcaps
6445     \PackageError{glossaries}{Option clash: 'smallcaps' and 'dua'
6446       can't both be set}{}%
6447   \else
6448     \ifglsacrsmaller
6449       \PackageError{glossaries}{Option clash: 'smaller' and 'dua'
6450         can't both be set}{}%
6451     \fi

```

```

6452 \fi
6453 \renewcommand{\newacronym}[4][\]{%
6454   \ifx\@glsacronymlists\@empty
6455     \def\@glo@type{\acronymtype}%
6456     \setkeys{glossentry}{##1}%
6457     \DeclareAcronymList{\@glo@type}%
6458     \SetDescriptionDUAAcronymDisplayStyle{\@glo@type}%
6459   \fi
6460   \glskeylisttok{##1}%
6461   \glslabeltok{##2}%
6462   \glsshorttok{##3}%
6463   \gslongtok{##4}%
6464   \newacronymhook
6465   \DescriptionDUANewAcronymDef
6466 }%

  Set display.
6467 \@for\@gls@type:=\@glsacronymlists\do{%
6468   \SetDescriptionDUAAcronymDisplayStyle{\@gls@type}%
6469 }%
6470 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary using the description setting (but not footnote or dua).

```

6471 \newcommand*\SetDescriptionAcronymDisplayStyle}[1]{%
6472   \def\glsentryfmt[#1]{%

6473     \ifdefempty\glscustomtext
6474     {%
6475       \ifglsused{\glslabel}%
6476       {%

  Move the inserted text outside of \acronymfont
6477         \let\gls@org@insert\glsinsert
6478         \let\glsinsert\@empty
6479         \acronymfont{\glsentryfmt}\gls@org@insert
6480       }%
6481     {%
6482       \glsentryfmt
6483       \ifglsymbol{\glslabel}%
6484       {%
6485         \glsifplural
6486         {%
6487           \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
6488         }%
6489       {%
6490         \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
6491       }%
6492       \space\protect\firstacronymfont
6493       {\glsacronymfont}

```

```

6494         {\@glo@symbol}
6495         {\@glo@symbol}
6496         {\mfirstucMakeUppercase{\@glo@symbol}}})%
6497     }%
6498     {}%
6499 }%
6500 }%
6501 {\glscustomtext\glsinsert}%
6502 }%
6503 }

```

ptionNewAcronymDef

```

6504 \newcommand*{\DescriptionNewAcronymDef}{%
6505   \edef\@do@newglossaryentry{%
6506     \noexpand\newglossaryentry{\the\glslabeltok}%
6507     {%
6508       type=\acronymtype,%
6509       name={\noexpand
6510         \acrnameformat{\the\glsshorttok}{\the\glslongtok}},%
6511       sort={\the\glsshorttok},%
6512       first={\the\glslongtok},%
6513       firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6514       text={\the\glsshorttok},%
6515       plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6516       short={\the\glsshorttok},%
6517       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6518       long={\the\glslongtok},%
6519       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6520       symbol={\noexpand\@glo@text},%
6521       symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6522       \the\glskeylisttok}%
6523   }%
6524   \let\@org@gls@assign@firstpl\gls@assign@firstpl
6525   \let\@org@gls@assign@plural\gls@assign@plural
6526   \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6527   \def\gls@assign@firstpl##1##2{%
6528     \@@gls@expand@field{##1}{firstpl}{##2}%
6529   }%
6530   \def\gls@assign@plural##1##2{%
6531     \@@gls@expand@field{##1}{plural}{##2}%
6532   }%
6533   \def\gls@assign@symbolplural##1##2{%
6534     \@@gls@expand@field{##1}{symbolplural}{##2}%
6535   }%
6536   \@do@newglossaryentry
6537   \let\gls@assign@firstpl\@org@gls@assign@firstpl
6538   \let\gls@assign@plural\@org@gls@assign@plural
6539   \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6540 }

```

riptionAcronymStyle Option description is used, but not dua or footnote. Store long form in first key and short form in text and symbol key. The name is stored using `\acrnameformat` to allow the user to override the way the name is displayed in the list of acronyms.

```

6541 \newcommand*\SetDescriptionAcronymStyle}{%
6542   \renewcommand{\newacronym}[4][]{%
6543     \ifx\@glsacronymlists\@empty
6544       \def\@glo@type{\acronymtype}%
6545       \setkeys{glossentry}{##1}%
6546       \DeclareAcronymList{\@glo@type}%
6547       \SetDescriptionAcronymDisplayStyle{\@glo@type}%
6548     \fi
6549     \glskeylisttok{##1}%
6550     \glslabeltok{##2}%
6551     \glsshorttok{##3}%
6552     \gslongtok{##4}%
6553     \newacronymhook
6554     \DescriptionNewAcronymDef
6555   }%

```

Set display.

```

6556   \@for\@gls@type:=\@glsacronymlists\do{%
6557     \SetDescriptionAcronymDisplayStyle{\@gls@type}%
6558   }%

```

Redefine `\acronymfont` if small caps required. The plural suffix is set in an upright font so that it remains in normal lower case, otherwise it looks as though it's part of the acronym.

```

6559   \ifglsacrsmallcaps
6560     \renewcommand{\acronymfont}[1]{\textsc{##1}}
6561     \renewcommand*\acrpluralsuffix}{%
6562       \glstextup{\glspluralsuffix}}%
6563   \else
6564     \ifglsacrsmaller
6565       \renewcommand*\acronymfont}[1]{\textsmaller{##1}}%
6566     \fi
6567   \fi
6568 }%

```

AcronymDisplayStyle Sets the acronym display style for given glossary with footnote setting (but not description or dua).

```

6569 \newcommand*\SetFootnoteAcronymDisplayStyle}[1]{%
6570   \defglsentryfmt[#1]{%

6571     \ifdefempty\glscustomtext
6572     {%

```

Move the inserted text outside of `\acronymfont`

```

6573     \let\gls@org@insert\glsinsert

```

```

6574 \let\glsinsert\@empty
6575 \ifglsused{\glslabel}%
6576 {%
6577 \acronymfont{\glsgetentryfmt}\gls@org@insert
6578 }%
6579 {%
6580 \firstacronymfont{\glsgetentryfmt}\gls@org@insert
6581 \ifglsashaslong{\glslabel}%
6582 {%
6583 \expandafter\protect\expandafter\acrfootnote\expandafter
6584 {\@gls@link@opts}{\@gls@link@label}%
6585 {%
6586 \glsifplural
6587 {\glsentrylongpl{\glslabel}}%
6588 {\glsentrylong{\glslabel}}%
6589 }%
6590 }%
6591 {}%
6592 }%
6593 }%
6594 {\glscustomtext\glsinsert}%
6595 }%
6596 }

```

otnoteNewAcronymDef

```

6597 \newcommand*{\FootnoteNewAcronymDef}{%
6598 \edef\@do@newglossaryentry{%
6599 \noexpand\newglossaryentry{\the\glslabeltok}%
6600 {%
6601 type=\acronymtype,%
6602 name={\noexpand\acronymfont{\the\glsshorttok}},%
6603 sort={\the\glsshorttok},%
6604 text={\the\glsshorttok},%
6605 plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6606 first={\the\glsshorttok},%
6607 firstplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6608 short={\the\glsshorttok},%
6609 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6610 long={\the\glslongtok},%
6611 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6612 description={\the\glslongtok},%
6613 descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6614 \the\glskeylisttok
6615 }%
6616 }%
6617 \let\@org@gls@assign@plural\gls@assign@plural
6618 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6619 \let\@org@gls@assign@descplural\gls@assign@descplural
6620 \def\gls@assign@firstpl##1##2{%

```

```

6621 \@@gls@expand@field{##1}{firstpl}{##2}%
6622 }%
6623 \def\gls@assign@plural##1##2{%
6624 \@@gls@expand@field{##1}{plural}{##2}%
6625 }%
6626 \def\gls@assign@descplural##1##2{%
6627 \@@gls@expand@field{##1}{descplural}{##2}%
6628 }%
6629 \do@newglossaryentry
6630 \let\gls@assign@plural\@org@gls@assign@plural
6631 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6632 \let\gls@assign@descplural\@org@gls@assign@descplural
6633 }

```

`footnoteAcronymStyle` If footnote package option is specified, set the first use to append the long form (stored in description) as a footnote. Use the description key to store the long form.

```

6634 \newcommand*\SetFootnoteAcronymStyle{%
6635 \renewcommand{\newacronym}[4][ ]{%
6636 \ifx\@glsacronymlists\@empty
6637 \def\@glo@type{\acronymtype}%
6638 \setkeys{glossentry}{##1}%
6639 \DeclareAcronymList{\@glo@type}%
6640 \SetFootnoteAcronymDisplayStyle{\@glo@type}%
6641 \fi
6642 \glskeylisttok{##1}%
6643 \glslabeltok{##2}%
6644 \glsshorttok{##3}%
6645 \glslongtok{##4}%
6646 \newacronymhook
6647 \FootnoteNewAcronymDef
6648 }%

```

Set display

```

6649 \@for\@gls@type:=\@glsacronymlists\do{%
6650 \SetFootnoteAcronymDisplayStyle{\@gls@type}%
6651 }%

```

Redefine `\acronymfont` if small caps required. The plural suffix is set in an upright font so that it remains in normal lower case, otherwise it looks as though it's part of the acronym.

```

6652 \ifglsacrsmallcaps
6653 \renewcommand*\acronymfont}[1]{\textsc{##1}}%
6654 \renewcommand*\acrpluralsuffix{%
6655 \glstextup{\glspluralsuffix}}%
6656 \else
6657 \ifglsacrsmaller
6658 \renewcommand*\acronymfont}[1]{\textsmaller{##1}}%
6659 \fi
6660 \fi

```

Check for option clash

```
6661 \ifglsacrdua
6662   \PackageError{glossaries}{Option clash: ‘footnote’ and ‘dua’
6663     can’t both be set}{}%
6664 \fi
6665 }%
```

`\dsdoparenifnotempty` Do a space followed by the argument if the argument doesn't expand to empty or `\relax`. If argument isn't empty (or `\relax`), apply the macro to it given in the second argument.

```
6666 \DeclareRobustCommand*{\glsdoparenifnotempty}[2]{%
6667   \protected@edef\gls@tmp{#1}%
6668   \ifdefempty\gls@tmp
6669     {}%
6670     {%
6671       \ifx\gls@tmp\@gls@default@value
6672         \else
6673           \space (#2{#1})%
6674         \fi
6675     }%
6676 }
```

`AcronymDisplayStyle` Sets the acronym display style for given glossary where neither footnote nor description is required, but smallcaps or smaller specified.

```
6677 \newcommand*{\SetSmallAcronymDisplayStyle}[1]{%
6678   \defglsentryfmt[#1]{%
6679     \ifdefempty\glscustomtext
6680       {%
```

Move the inserted text outside of `\acronymfont`

```
6681     \let\gls@org@insert\glsinsert
6682     \let\glsinsert\@empty
6683     \ifglsused{\glslabel}%
6684     {%
6685       \acronymfont{\glsgenentryfmt}\gls@org@insert
6686     }%
6687     {%
6688       \glsgenentryfmt
6689       \ifglsymbol{\glslabel}%
6690       {%
6691         \glsifplural
6692         {%
6693           \def\@glo@symbol{\glsentrysymbolplural{\glslabel}}%
6694         }%
6695         {%
6696           \def\@glo@symbol{\glsentrysymbol{\glslabel}}%
6697         }%
6698         \space
```



```

6699         (\glscapscase
6700         {\firstacronymfont{\@glo@symbol}}%
6701         {\firstacronymfont{\@glo@symbol}}%
6702         {\firstacronymfont{\mfirstucMakeUppercase{\@glo@symbol}}})%
6703     }%
6704     {}%
6705 }%
6706 }%
6707 {\glscustomtext\glsinsert}%
6708 }%
6709 }

```

\SmallNewAcronymDef

```

6710 \newcommand*\SmallNewAcronymDef{%
6711   \edef\@do@newglossaryentry{%
6712     \noexpand\newglossaryentry{\the\glslabeltok}%
6713     {%
6714       type=\acronymtype,%
6715       name={\noexpand\acronymfont{\the\glsshorttok}},%
6716       sort={\the\glsshorttok},%
6717       text={\the\glsshorttok},%

```

Default to the short plural.

```

6718     plural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6719     first={\the\glslongtok},%

```

Default to the long plural.

```

6720     firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6721     short={\the\glsshorttok},%
6722     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6723     long={\the\glslongtok},%
6724     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6725     description={\noexpand\@glo@first},%
6726     descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6727     symbol={\the\glsshorttok},%

```

Default to the short plural.

```

6728     symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6729     \the\glskeylisttok
6730 }%
6731 }%
6732 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6733 \let\@org@gls@assign@plural\gls@assign@plural
6734 \let\@org@gls@assign@descplural\gls@assign@descplural
6735 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6736 \def\gls@assign@firstpl##1##2{%
6737   \@@gls@expand@field{##1}{firstpl}{##2}%
6738 }%
6739 \def\gls@assign@plural##1##2{%
6740   \@@gls@expand@field{##1}{plural}{##2}%

```

```

6741 }%
6742 \def\gls@assign@descplural##1##2{%
6743   \@@gls@expand@field{##1}{descplural}{##2}%
6744 }%
6745 \def\gls@assign@symbolplural##1##2{%
6746   \@@gls@expand@field{##1}{symbolplural}{##2}%
6747 }%
6748 \do@newglossaryentry
6749 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6750 \let\gls@assign@plural\@org@gls@assign@plural
6751 \let\gls@assign@descplural\@org@gls@assign@descplural
6752 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6753 }

```

`etSmallAcronymStyle` Neither footnote nor description required, but smallcaps or smaller specified.

Use the symbol key to store the short form and first to store the long form.

```

6754 \newcommand*\SetSmallAcronymStyle{%
6755   \renewcommand{\newacronym}[4][]{%
6756     \ifx\@glsacronymlists\@empty
6757       \def\@glo@type{\acronymtype}%
6758       \setkeys{glossentry}{##1}%
6759       \DeclareAcronymList{\@glo@type}%
6760       \SetSmallAcronymDisplayStyle{\@glo@type}%
6761     \fi
6762     \glskeylisttok{##1}%
6763     \glslabeltok{##2}%
6764     \glsshorttok{##3}%
6765     \glslongtok{##4}%
6766     \newacronymhook
6767     \SmallNewAcronymDef
6768   }%

```

Change the display since first only contains long form.

```

6769 \@for\@gls@type:=\@glsacronymlists\do{%
6770   \SetSmallAcronymDisplayStyle{\@gls@type}%
6771 }%

```

Redefine `\acronymfont` if small caps required. The plural suffix is set in an upright font so that it remains in normal lower case, otherwise it looks as though it's part of the acronym.

```

6772 \ifglsacrsmallcaps
6773   \renewcommand*\acronymfont}[1]{\textsc{##1}}
6774   \renewcommand*\acrpluralsuffix{%
6775     \glstextup{\glspluralsuffix}}%
6776 \else
6777   \renewcommand*\acronymfont}[1]{\textsmaller{##1}}
6778 \fi

```

check for option clash

```

6779 \ifglsacrdua

```

```

6780 \ifglsacrsmallcaps
6781 \PackageError{glossaries}{Option clash: ‘smallcaps’ and ‘dua’
6782 can’t both be set}{}%
6783 \else
6784 \PackageError{glossaries}{Option clash: ‘smaller’ and ‘dua’
6785 can’t both be set}{}%
6786 \fi
6787 \fi
6788 }%

```

`\SetDUADisplayStyle` Sets the acronym display style for given glossary with dua setting.

```

6789 \newcommand*\SetDUADisplayStyle[1]{%
6790 \defglsentryfmt[#1]{\glsgenentryfmt}%
6791 }

```

`\DUANewAcronymDef`

```

6792 \newcommand*\DUANewAcronymDef{%
6793 \edef\do@newglossaryentry{%
6794 \noexpand\newglossaryentry{\the\glslabeltok}%
6795 {%
6796 type=\acronymtype,%
6797 name={\the\glsshorttok},%
6798 text={\the\glslongtok},%
6799 first={\the\glslongtok},%
6800 plural={\noexpand\expandonce\noexpand\@glo@longpl},%
6801 firstplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6802 short={\the\glsshorttok},%
6803 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6804 long={\the\glslongtok},%
6805 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6806 description={\the\glslongtok},%
6807 descriptionplural={\noexpand\expandonce\noexpand\@glo@longpl},%
6808 symbol={\the\glsshorttok},%
6809 symbolplural={\noexpand\expandonce\noexpand\@glo@shortpl},%
6810 \the\glskeylisttok
6811 }%
6812 }%
6813 \let\@org@gls@assign@firstpl\gls@assign@firstpl
6814 \let\@org@gls@assign@plural\gls@assign@plural
6815 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
6816 \let\@org@gls@assign@descplural\gls@assign@descplural
6817 \def\gls@assign@firstpl##1##2{%
6818 \@@gls@expand@field{##1}{firstpl}{##2}%
6819 }%
6820 \def\gls@assign@plural##1##2{%
6821 \@@gls@expand@field{##1}{plural}{##2}%
6822 }%
6823 \def\gls@assign@symbolplural##1##2{%
6824 \@@gls@expand@field{##1}{symbolplural}{##2}%

```

```

6825 }%
6826 \def\gls@assign@descplural##1##2{%
6827   \@@gls@expand@field{##1}{descplural}{##2}%
6828 }%
6829 \do@newglossaryentry
6830 \let\gls@assign@firstpl\@org@gls@assign@firstpl
6831 \let\gls@assign@plural\@org@gls@assign@plural
6832 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
6833 \let\gls@assign@descplural\@org@gls@assign@descplural
6834 }

```

`\SetDUASyle` Always expand acronyms.

```

6835 \newcommand*\SetDUASyle{%
6836   \renewcommand{\newacronym}[4] []{%
6837     \ifx\@glsacronymlists\@empty
6838       \def\@glo@type{\acronymtype}%
6839       \setkeys{glossentry}{##1}%
6840       \DeclareAcronymList{\@glo@type}%
6841       \SetDUADisplayStyle{\@glo@type}%
6842       \fi
6843       \glskeylisttok{##1}%
6844       \glslabeltok{##2}%
6845       \glsshorttok{##3}%
6846       \glslongtok{##4}%
6847       \newacronymhook
6848       \DUANewAcronymDef
6849     }%
6850   \@for\@gls@type:=\@glsacronymlists\do{%
6851     \SetDUADisplayStyle{\@gls@type}%
6852   }%
6853 }

```

Set the display

`\SetAcronymStyle`

```

6854 \newcommand*\SetAcronymStyle{%
6855   \SetDefaultAcronymStyle
6856   \ifglsacrdescription
6857     \ifglsacrfootnote
6858       \SetDescriptionFootnoteAcronymStyle
6859     \else
6860       \ifglsacrdua
6861         \SetDescriptionDUAAcronymStyle
6862       \else
6863         \SetDescriptionAcronymStyle
6864       \fi
6865     \fi
6866   \else
6867     \ifglsacrfootnote
6868       \SetFootnoteAcronymStyle

```

```

6869 \else
6870 \ifthenelse{\boolean{glsacrsmallcaps}}\OR
6871 \boolean{glsacrsmaller}}%
6872 {%
6873 \SetSmallAcronymStyle
6874 }%
6875 {%
6876 \ifglsacrdua
6877 \SetDUASStyle
6878 \fi
6879 }%
6880 \fi
6881 \fi
6882 }

```

Set the acronym style according to the package options

```
6883 \SetAcronymStyle
```

Allow user to define their own custom acronyms. (For compatibility with versions before v3.0, the short form is stored in the user1 key, the plural short form is stored in the user2 key, the long form is stored in the user3 key and the plural long form is stored in the user4 key.) Defaults to displaying only the acronym with the long form as the description.

`\SetCustomDisplayStyle` Sets the acronym display style.

```

6884 \newcommand*\SetCustomDisplayStyle[1]{%
6885 \defglsentryfmt[#1]{\glsgenentryfmt}%
6886 }

```

`\CustomAcronymFields`

```

6887 \newcommand*\CustomAcronymFields{%
6888 name={\the\glsshorttok},%
6889 description={\the\glslongtok},%
6890 first={\acrfullformat{\the\glslongtok}{\the\glsshorttok}},%
6891 firstplural={\acrfullformat
6892 {\noexpand\glsentrylongpl{\the\glslabeltok}}%
6893 {\noexpand\glsentryshortpl{\the\glslabeltok}}},%
6894 text={\the\glsshorttok},%
6895 plural={\the\glsshorttok\noexpand\acrpluralsuffix}%
6896 }

```

`\CustomNewAcronymDef`

```

6897 \newcommand*\CustomNewAcronymDef{%
6898 \protected@edef\do@newglossaryentry{%
6899 \noexpand\newglossaryentry{\the\glslabeltok}%
6900 {%
6901 type=\acronymtype,%
6902 short={\the\glsshorttok},%

```

```

6903     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
6904     long={\the\glslongtok},%
6905     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
6906     user1={\the\glsshorttok},%
6907     user2={\the\glsshorttok\noexpand\acrpluralsuffix},%
6908     user3={\the\glslongtok},%
6909     user4={\the\glslongtok\noexpand\acrpluralsuffix},%
6910     \CustomAcronymFields,%
6911     \the\glskeylisttok
6912   }%
6913 }%
6914 \@do@newglossaryentry
6915 }

```

\SetCustomStyle

```

6916 \newcommand*{\SetCustomStyle}{%
6917   \renewcommand{\newacronym}[4][[]]{%
6918     \ifx\@glsacronymlists\@empty
6919       \def\@glo@type{\acronymtype}%
6920       \setkeys{glossentry}{##1}%
6921       \DeclareAcronymList{\@glo@type}%
6922       \SetCustomDisplayStyle{\@glo@type}%
6923     \fi
6924     \glskeylisttok{##1}%
6925     \glslabeltok{##2}%
6926     \glsshorttok{##3}%
6927     \glslongtok{##4}%
6928     \newacronymhook
6929     \CustomNewAcronymDef
6930   }%
6931   Set the display
6932   \@for\@gls@type:=\@glsacronymlists\do{%
6933     \SetCustomDisplayStyle{\@gls@type}%
6934 }%

```

1.18 Predefined Glossary Styles

The glossaries bundle comes with some predefined glossary styles. These need to be loaded now for the style option to use them.

First, the glossary hyper-navigation commands need to be loaded.

```
6935 \RequirePackage{glossary-hypernav}
```

The styles that use list-like environments. These are not loaded if the `nolist` option is used:

```
6936 \@gls@loadlist
```

The styles that use the `longtable` environment. These are not loaded if the `no-long` package option is used.

```
6937 \@gls@loadlong
```

The styles that use the supertabular environment. These are not loaded if the nosuper package option is used or if the package isn't installed.

```
6938 \@gls@loadsuper
```

The tree-like styles. These are not loaded if the notree package option is used.

```
6939 \@gls@loadtree
```

The default glossary style is set according to the style package option, but can be overridden by \glossarystyle. The required style must be defined at this point.

```
6940 \ifx\@glossary@default@style\relax
```

```
6941 \else
```

```
6942 \setglossarystyle{\@glossary@default@style}
```

```
6943 \fi
```

1.19 Debugging Commands

```
\showgloparent \showgloparent{<label>}
```

```
6944 \newcommand*\showgloparent}[1]{%
```

```
6945 \expandafter\show\csname glo@glstetoklabel{#1}@parent\endcsname
```

```
6946 }
```

```
\showglolevel \showglolevel{<label>}
```

```
6947 \newcommand*\showglolevel}[1]{%
```

```
6948 \expandafter\show\csname glo@glstetoklabel{#1}@level\endcsname
```

```
6949 }
```

```
\showglotext \showglotext{<label>}
```

```
6950 \newcommand*\showglotext}[1]{%
```

```
6951 \expandafter\show\csname glo@glstetoklabel{#1}@text\endcsname
```

```
6952 }
```

```
\showgloplural \showgloplural{<label>}
```

```
6953 \newcommand*\showgloplural}[1]{%
```

```
6954 \expandafter\show\csname glo@glstetoklabel{#1}@plural\endcsname
```

```
6955 }
```

`\showglofirst` `\showglofirst{<label>}`

```
6956 \newcommand*{\showglofirst}[1]{%
6957   \expandafter\show\csname glo@glstetoklabel{#1}@first\endcsname
6958 }
```

`\showglofirstpl` `\showglofirstpl{<label>}`

```
6959 \newcommand*{\showglofirstpl}[1]{%
6960   \expandafter\show\csname glo@glstetoklabel{#1}@firstpl\endcsname
6961 }
```

`\showgloftype` `\showgloftype{<label>}`

```
6962 \newcommand*{\showgloftype}[1]{%
6963   \expandafter\show\csname glo@glstetoklabel{#1}@type\endcsname
6964 }
```

`\showglocounter` `\showglocounter{<label>}`

```
6965 \newcommand*{\showglocounter}[1]{%
6966   \expandafter\show\csname glo@glstetoklabel{#1}@counter\endcsname
6967 }
```

`\showglouserii` `\showglouserii{<label>}`

```
6968 \newcommand*{\showglouserii}[1]{%
6969   \expandafter\show\csname glo@glstetoklabel{#1}@userii\endcsname
6970 }
```

`\showglouserii` `\showglouserii{<label>}`

```
6971 \newcommand*{\showglouserii}[1]{%
6972   \expandafter\show\csname glo@glstetoklabel{#1}@userii\endcsname
6973 }
```


`\showglouseriii` `\showglouseriii{<label>}`

```
6974 \newcommand*{\showglouseriii}[1]{%
6975   \expandafter\show\csname glo@glsdetoklabel{#1}@useriii\endcsname
6976 }
```

`\showglouseriv` `\showglouseriv{<label>}`

```
6977 \newcommand*{\showglouseriv}[1]{%
6978   \expandafter\show\csname glo@glsdetoklabel{#1}@useriv\endcsname
6979 }
```

`\showglouserv` `\showglouserv{<label>}`

```
6980 \newcommand*{\showglouserv}[1]{%
6981   \expandafter\show\csname glo@glsdetoklabel{#1}@userv\endcsname
6982 }
```

`\showglouservi` `\showglouservi{<label>}`

```
6983 \newcommand*{\showglouservi}[1]{%
6984   \expandafter\show\csname glo@glsdetoklabel{#1}@uservi\endcsname
6985 }
```

`\showgloname` `\showgloname{<label>}`

```
6986 \newcommand*{\showgloname}[1]{%
6987   \expandafter\show\csname glo@glsdetoklabel{#1}@name\endcsname
6988 }
```

`\showglodesc` `\showglodesc{<label>}`

```
6989 \newcommand*{\showglodesc}[1]{%
6990   \expandafter\show\csname glo@glsdetoklabel{#1}@desc\endcsname
6991 }
```

`\showglodescplural` `\showglodescplural{<label>}`

```
6992 \newcommand*{\showglodescplural}[1]{%
6993   \expandafter\show\csname glo@glsdetoklabel{#1}@descplural\endcsname
6994 }
```

`\showglosort` `\showglosort{<label>}`

```
6995 \newcommand*{\showglosort}[1]{%
6996   \expandafter\show\csname glo@glsdetoklabel{#1}@sort\endcsname
6997 }
```

`\showglosymbol` `\showglosymbol{<label>}`

```
6998 \newcommand*{\showglosymbol}[1]{%
6999   \expandafter\show\csname glo@glsdetoklabel{#1}@symbol\endcsname
7000 }
```

`\showglosymbolplural` `\showglosymbolplural{<label>}`

```
7001 \newcommand*{\showglosymbolplural}[1]{%
7002   \expandafter\show\csname glo@glsdetoklabel{#1}@symbolplural\endcsname
7003 }
```

`\showgloshort` `\showgloshort{<label>}`

```
7004 \newcommand*{\showgloshort}[1]{%
7005   \expandafter\show\csname glo@glsdetoklabel{#1}@short\endcsname
7006 }
```

`\showglolong` `\showglolong{<label>}`

```
7007 \newcommand*{\showglolong}[1]{%
7008   \expandafter\show\csname glo@glsdetoklabel{#1}@long\endcsname
7009 }
```

`\showgloindex` `\showgloindex{<label>}`

```
7010 \newcommand*{\showgloindex}[1]{%
7011   \expandafter\show\csname glo@\glsdetoklabel{#1}@index\endcsname
7012 }
```

`\showgloflag` `\showgloflag{<label>}`

```
7013 \newcommand*{\showgloflag}[1]{%
7014   \expandafter\show\csname ifglo@\glsdetoklabel{#1}@flag\endcsname
7015 }
```

`\showgloclist` `\showgloclist{<label>}`

```
7016 \newcommand*{\showgloclist}[1]{%
7017   \expandafter\show\csname glo@\glsdetoklabel{#1}@loclist\endcsname
7018 }
```

`\showacronymlists` `\showacronymlists`

Show list of glossaries that have been flagged as a list of acronyms.

```
7019 \newcommand*{\showacronymlists}{%
7020   \show\@glsacronymlists
7021 }
```

`\showglossaries` `\showglossaries`

Show list of defined glossaries.

```
7022 \newcommand*{\showglossaries}{%
7023   \show\@glo@types
7024 }
```

`\showglossaryin` `\showglossaryin{<glossary-label>}`

Show the ‘in’ extension for the given glossary.

```
7025 \newcommand*{\showglossaryin}[1]{%
7026   \expandafter\show\csname @glo@type@#1@in\endcsname
7027 }
```

`\showglossaryout` `\showglossaryout{<glossary-label>}`

Show the ‘out’ extension for the given glossary.

```
7028 \newcommand*{\showglossaryout}[1]{%
7029   \expandafter\show\csname @glotype@#1@out\endcsname
7030 }
```

`\showglossarytitle` `\showglossarytitle{<glossary-label>}`

Show the title for the given glossary.

```
7031 \newcommand*{\showglossarytitle}[1]{%
7032   \expandafter\show\csname @glotype@#1@title\endcsname
7033 }
```

`\showglossarycounter` `\showglossarycounter{<glossary-label>}`

Show the counter for the given glossary.

```
7034 \newcommand*{\showglossarycounter}[1]{%
7035   \expandafter\show\csname @glotype@#1@counter\endcsname
7036 }
```

`\showglossaryentries` `\showglossaryentries{<glossary-label>}`

Show the list of entry labels for the given glossary.

```
7037 \newcommand*{\showglossaryentries}[1]{%
7038   \expandafter\show\csname glolist@#1\endcsname
7039 }
```

1.20 Compatibility with version 2.07 and below

In order to fix some bugs in v3.0, it was necessary to change the way information is written to the `glo` file, which also meant a change in the format of the Xindy style file. The compatibility option is meant for documents that use a customised Xindy style file with `\noist`. With the compatibility option, hopefully xindy will still be able to process the old document, but the bugs will remain. The issues in versions 2.07 and below:

- With xindy, the counter used by the entry was hard-coded into the Xindy style file. This meant that you couldn't use the counter to swap counters.
- With both xindy and makeindex, if used with hyperref and `\theH{<counter>}` was different to `\thecounter`, the link in the location number would be undefined.

```

7040 \csname ifglscompatible-2.07\endcsname
7041 \RequirePackage{glossaries-compatible-207}
7042 \fi

```

2 Prefix Support (glossaries-prefix Code)

This package provides a means of adding prefixes to your glossary entries. For example, you may want to use “a \gls{<label>}” on first use but use “an \gls{<label>}” on subsequent use.

```

7043 \NeedsTeXFormat{LaTeX2e}
7044 \ProvidesPackage{glossaries-prefix}[2014/07/30 v4.08 (NLCT)]

  Pass all options to glossaries:
7045 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}

  Process options:
7046 \ProcessOptions

  Load glossaries:
7047 \RequirePackage{glossaries}

  Add the new keys:
7048 \define@key{glossentry}{prefixfirst}{\def\@glo@entryprefixfirst{#1}}%
7049 \define@key{glossentry}{prefixfirstplural}{\def\@glo@entryprefixfirstplural{#1}}%
7050 \define@key{glossentry}{prefix}{\def\@glo@entryprefix{#1}}%
7051 \define@key{glossentry}{prefixplural}{\def\@glo@entryprefixplural{#1}}%

  Add them to \@gls@keymap:
7052 \appto\@gls@keymap{,%
7053   {prefixfirst}{prefixfirst},%
7054   {prefixfirstplural}{prefixfirstplural},%
7055   {prefix}{prefix},%
7056   {prefixplural}{prefixplural}}%
7057 }

  Set the default values:
7058 \appto\@newglossaryentryprehook{%
7059   \def\@glo@entryprefix{}%
7060   \def\@glo@entryprefixplural{}%
7061   \let\@glo@entryprefixfirst\@gls@default@value
7062   \let\@glo@entryprefixfirstplural\@gls@default@value
7063 }

  Set the assignment code:
7064 \appto\@newglossaryentryposthook{%
7065   \gls@assign@field{\@glo@label}{prefix}{\@glo@entryprefix}%
7066   \gls@assign@field{\@glo@label}{prefixplural}{\@glo@entryprefixplural}}%

  If prefixfirst has not been supplied, make it the same as prefix.
7067 \expandafter\gls@assign@field\expandafter
7068   {\csname glo@\@glo@label @prefix\endcsname}{\@glo@label}{prefixfirst}%
7069   {\@glo@entryprefixfirst}%

```

If prefixfirstplural has not been supplied, make it the same as prefixplural.

```

7070 \expandafter\gls@assign@field\expandafter
7071   {\csname glo@\@glo@label @prefixplural\endcsname}{\@glo@label}%
7072   {prefixfirstplural}{\@glo@entryprefixfirstplural}%
7073 }

```

Define commands to access these fields:

glsentryprefixfirst

```
7074 \newcommand*\glsentryprefixfirst[1]{\csuse{glo@#1@prefixfirst}}
```

ryprefixfirstplural

```
7075 \newcommand*\glsentryprefixfirstplural[1]{\csuse{glo@#1@prefixfirstplural}}
```

\glsentryprefix

```
7076 \newcommand*\glsentryprefix[1]{\csuse{glo@#1@prefix}}
```

lentryprefixplural

```
7077 \newcommand*\glsentryprefixplural[1]{\csuse{glo@#1@prefixplural}}
```

Now for the initial upper case variants:

Glsentryprefixfirst

```

7078 \newrobustcmd*\Glsentryprefixfirst[1]{%
7079   \protected@edef\@glo@text{\csname glo@#1@prefixfirst\endcsname}%
7080   \xmakefirstuc\@glo@text
7081 }

```

ryprefixfirstplural

```

7082 \newrobustcmd*\Glsentryprefixfirstplural[1]{%
7083   \protected@edef\@glo@text{\csname glo@#1@prefixfirstplural\endcsname}%
7084   \xmakefirstuc\@glo@text
7085 }

```

\Glsentryprefix

```

7086 \newrobustcmd*\Glsentryprefix[1]{%
7087   \protected@edef\@glo@text{\csname glo@#1@prefix\endcsname}%
7088   \xmakefirstuc\@glo@text
7089 }

```

lentryprefixplural

```

7090 \newrobustcmd*\Glsentryprefixplural[1]{%
7091   \protected@edef\@glo@text{\csname glo@#1@prefixplural\endcsname}%
7092   \xmakefirstuc\@glo@text
7093 }

```

Define commands to determine if the prefix keys have been set:

`\ifglshasprefix`

```
7094 \newcommand*\ifglshasprefix}[3]{%
7095   \ifcempty{glo@#1@prefix}%
7096   {#3}%
7097   {#2}%
7098 }
```

`ifglshasprefixplural`

```
7099 \newcommand*\ifglshasprefixplural}[3]{%
7100   \ifcempty{glo@#1@prefixplural}%
7101   {#3}%
7102   {#2}%
7103 }
```

`ifglshasprefixfirst`

```
7104 \newcommand*\ifglshasprefixfirst}[3]{%
7105   \ifcempty{glo@#1@prefixfirst}%
7106   {#3}%
7107   {#2}%
7108 }
```

`asprefixfirstplural`

```
7109 \newcommand*\ifglshasprefixfirstplural}[3]{%
7110   \ifcempty{glo@#1@prefixfirstplural}%
7111   {#3}%
7112   {#2}%
7113 }
```

Define commands that insert the prefix before commands like `\gls`:

`\pgls`

```
7114 \newrobustcmd{\pgls}{\@gls@hyp@opt\@pgls}
```

`\@pgls` Unstarred version.

```
7115 \newcommand*\@pgls}[2][ ]{%
7116   \new@ifnextchar[%
7117     {\@pgls@{#1}{#2}}%
7118     {\@pgls@{#1}{#2}[ ]}%
7119 }
```

`\@pgls@` Read in the final optional argument:

```
7120 \def\@pgls@#1#2[#3]{%
7121   \glsdoifexists{#2}%
7122   {%
7123     \ifglsused{#2}%
7124     {%
7125       \glsentryprefix{#2}%
7126     }%

```

```

7127   {%
7128     \glsentryprefixfirst{#2}%
7129   }%
7130   \@gls@{#1}{#2}[#3]%
7131 }%
7132 }

```

Similarly for the plural version:

```

\pglsp1
7133 \newrobustcmd{\pglsp1}{\@gls@hyp@opt\@pglsp1}

```

\@pglsp1 Unstarred version.

```

7134 \newcommand*{\@pglsp1}[2] [] {%
7135   \new@ifnextchar [%
7136   {\@pglsp1@{#1}{#2}}%
7137   {\@pglsp1@{#1}{#2} []}%
7138 }

```

\@pglsp1@ Read in the final optional argument:

```

7139 \def\@pglsp1@#1#2[#3] {%
7140   \glsdoifexists{#2}%
7141   {%
7142     \ifglsused{#2}%
7143     {%
7144       \glsentryprefixplural{#2}%
7145     }%
7146     {%
7147       \glsentryprefixfirstplural{#2}%
7148     }%
7149     \@glspl@{#1}{#2}[#3]%
7150   }%
7151 }

```

Now for the first letter upper case versions:

```

\Pgls
7152 \newrobustcmd{\Pgls}{\@gls@hyp@opt\@Pgls}

```

\@Pgls Unstarred version.

```

7153 \newcommand*{\@Pgls}[2] [] {%
7154   \new@ifnextchar [%
7155   {\@Pgls@{#1}{#2}}%
7156   {\@Pgls@{#1}{#2} []}%
7157 }

```


\@Pgl@s@ Read in the final optional argument:

```
7158 \def\@Pgl@s@#1#2[#3]{%
7159   \glsdoifexists{#2}%
7160   {%
7161     \ifglsused{#2}%
7162     {%
7163       \ifglshasprefix{#2}%
7164       {%
7165         \Glsentryprefix{#2}%
7166         \@gls@{#1}{#2}[#3]%
7167       }%
7168       {\@Gls@{#1}{#2}[#3]}%
7169     }%
7170   }%
7171   \ifglshasprefixfirst{#2}%
7172   {%
7173     \Glsentryprefixfirst{#2}%
7174     \@gls@{#1}{#2}[#3]%
7175   }%
7176   {\@Gls@{#1}{#2}[#3]}%
7177 }%
7178 }%
7179 }
```

Similarly for the plural version:

\Pgl spl

```
7180 \newrobustcmd{\Pgl spl}{\@gls@hyp@opt\@Pgl spl}
```

\@Pgl spl Unstarred version.

```
7181 \newcommand*{\@Pgl spl}[2][ ]{%
7182   \new@ifnextchar[%
7183     {\@Pgl spl@{#1}{#2}}%
7184     {\@Pgl spl@{#1}{#2}[]}%
7185 }
```

\@Pgl spl@ Read in the final optional argument:

```
7186 \def\@Pgl spl@#1#2[#3]{%
7187   \glsdoifexists{#2}%
7188   {%
7189     \ifglsused{#2}%
7190     {%
7191       \ifglshasprefixplural{#2}%
7192       {%
7193         \Glsentryprefixplural{#2}%
7194         \@glspl@{#1}{#2}[#3]%
7195       }%
7196       {\@Glspl@{#1}{#2}[#3]}%

```

```

7197 }%
7198 {%
7199   \ifglshasprefixfirstplural{#2}%
7200   {%
7201     \Glsentryprefixfirstplural{#2}%
7202     \@glsp1@{#1}{#2}[#3]%
7203   }%
7204   {\@Glspl@{#1}{#2}[#3]}%
7205 }%
7206 }%
7207 }

```

Finally the all upper case versions:

`\PGLS`

```
7208 \newrobustcmd{\PGLS}{\@gls@hyp@opt\PGLS}
```

`\@PGLS` Unstarred version.

```

7209 \newcommand*{\@PGLS}[2][ ]{%
7210   \new@ifnextchar[%
7211     {\@PGLS@{#1}{#2}}%
7212     {\@PGLS@{#1}{#2}[ ]}%
7213 }

```

`\@PGLS@` Read in the final optional argument:

```

7214 \def\@PGLS@#1#2[#3]{%
7215   \glsdoifexists{#2}%
7216   {%
7217     \ifglsused{#2}%
7218     {%
7219       \mfirstucMakeUppercase{\glsentryprefix{#2}}%
7220     }%
7221     {%
7222       \mfirstucMakeUppercase{\glsentryprefixfirst{#2}}%
7223     }%
7224     \@Gls@{#1}{#2}[#3]%
7225   }%
7226 }

```

Plural version:

`\PGLSp1`

```
7227 \newrobustcmd{\PGLSp1}{\@gls@hyp@opt\PGLSp1}
```

`\@PGLSp1` Unstarred version.

```

7228 \newcommand*{\@PGLSp1}[2][ ]{%
7229   \new@ifnextchar[%
7230     {\@PGLSp1@{#1}{#2}}%

```

```
7231 {\@PGLSp1@{#1}{#2} []}%
7232 }
```

\@PGLSp1@ Read in the final optional argument:

```
7233 \def\@PGLSp1@#1#2[#3]{%
7234   \glsdoifexists{#2}%
7235   {%
7236     \ifglsused{#2}%
7237     {%
7238       \mfirstucMakeUppercase{\glsentryprefixplural{#2}}%
7239     }%
7240     {%
7241       \mfirstucMakeUppercase{\glsentryprefixfirstplural{#2}}%
7242     }%
7243     \@GLSp1@{#1}{#2}[#3]%
7244   }%
7245 }
```

3 Mfirstuc Documented Code

```
7246 \NeedsTeXFormat{LaTeX2e}
7247 \ProvidesPackage{mfirstuc}[2014/07/30 v1.09 (NLCT)]
```

Requires etoolbox:

```
7248 \RequirePackage{etoolbox}
```

\makefirstuc Syntax:

```
\makefirstuc{<text>}
```

Makes the first letter uppercase, but will skip initial control sequences if they are followed by a group and make the first thing in the group uppercase, unless the group is empty. Thus `\makefirstuc{abc}` will produce: `Abc`, `\makefirstuc{\ae bc}` will produce: `Æbc`, but `\makefirstuc{\emph{abc}}` will produce `Abc`. This is required by `\Gls` and `\Glspl`.

```
7249 \newif\if@glscs
7250 \newtoks\@glsmfirst
7251 \newtoks\@glsmrest
7252 \newrobustcmd*{\makefirstuc}[1]{%
7253   \def\gls@argi{#1}%
7254   \ifx\gls@argi\@empty
7255     \else
7256     \def\@gls@tmp{\ #1}%
7257     \@onelevel@sanitize\@gls@tmp
7258     \expandafter\@gls@checkcs\@gls@tmp\relax\relax
7259     \if@glscs
```

```

7260     \@gls@getbody #1{\@nil
7261     \ifx\@gls@rest\@empty
7262         \glsmakefirstuc{#1}%
7263     \else
7264         \expandafter\@gls@split\@gls@rest\@nil
7265         \ifx\@gls@first\@empty
7266             \glsmakefirstuc{#1}%
7267         \else
7268             \expandafter\@glsmfirst\expandafter{\@gls@first}%
7269             \expandafter\@glsmrest\expandafter{\@gls@rest}%
7270             \edef\@gls@domfirstuc{\noexpand\@gls@body
7271                 {\noexpand\glsmakefirstuc\the\@glsmfirst}%
7272                 \the\@glsmrest}%
7273             \@gls@domfirstuc
7274         \fi
7275     \fi
7276     \else
7277         \glsmakefirstuc{#1}%
7278     \fi
7279 \fi
7280 }

```

Put first argument in \@gls@first and second argument in \@gls@rest:

```

7281 \def\@gls@split#1#2\@nil{%
7282     \def\@gls@first{#1}\def\@gls@rest{#2}%
7283 }

7284 \def\@gls@checkcs#1 #2#3\relax{%
7285     \def\@gls@argi{#1}\def\@gls@argii{#2}%
7286     \ifx\@gls@argi\@gls@argii
7287         \@gls@csTRUE
7288     \else
7289         \@gls@csFALSE
7290     \fi
7291 }

```

\@gls@makefirstuc Make first thing upper case:

```
7292 \def\@gls@makefirstuc#1{\mfirstucMakeUppercase #1}
```

mfirstucMakeUppercase Allow user to replace \MakeUppercase with another case changing command.

```
7293 \newcommand*{\mfirstucMakeUppercase}{\MakeUppercase}
```

\glsmakefirstuc Provide a user command to make it easier to customise.

```
7294 \newcommand*{\glsmakefirstuc}[1]{\@gls@makefirstuc{#1}}
```

Get the first grouped argument and store in \@gls@body.

```
7295 \def\@gls@getbody#1#\def\@gls@body{#1}\@gls@gobbletonil}
```

Scoup up everything to \@nil and store in \@gls@rest:

```
7296 \def\@gls@gobbletonil#1\@nil{\def\@gls@rest{#1}}
```

`\xmakefirstuc` Expand argument once before applying `\makefirstuc` (added v1.01).

```
7297 \newcommand*\xmakefirstuc}[1]{%
7298 \expandafter\makefirstuc\expandafter{#1}}
```

`\capitalisewords` Capitalise each word in the argument. Words are considered to be separated by plain spaces (i.e. non-breakable spaces won't be considered a word break).

```
7299 \newrobustcmd*\capitalisewords}[1]{%
7300 \def\gls@add@space{}%
7301 \let\@mfu@domakefirstuc\makefirstuc
7302 \let\@mfu@checkword\@gobble
7303 \mfu@capitalisewords#1 \@nil\mfu@endcap
7304 }

7305 \def\mfu@capitalisewords#1 #2\mfu@endcap{%
7306 \def\mfu@cap@first{#1}%
7307 \def\mfu@cap@second{#2}%
7308 \gls@add@space
7309 \@mfu@checkword{#1}%
7310 \@mfu@domakefirstuc{#1}%
7311 \def\gls@add@space{ }%
7312 \ifx\mfu@cap@second\@nnil
7313 \let\next@mfu@cap\mfu@noop
7314 \else
7315 \let\next@mfu@cap\mfu@capitalisewords
7316 \let\@mfu@checkword\mfu@checkword
7317 \fi
7318 \next@mfu@cap#2\mfu@endcap
7319 }
7320 \def\mfu@noop#1\mfu@endcap{}
```

`\mfu@checkword` Check if word should be capitalised.

```
7321 \newcommand*\mfu@checkword[1]{%
7322 \ifinlist{#1}{\@mfu@nocaplist}%
7323 {%
7324 \let\@mfu@domakefirstuc\@firstofone
7325 }%
7326 {%
7327 \let\@mfu@domakefirstuc\makefirstuc
7328 }%
7329 }
```

`\@mfu@nocaplist` List of words that shouldn't be capitalised.

```
7330 \newcommand*\@mfu@nocaplist{}
```

`\MFUnocap` Provide the user with a means to add a word to the list.

```
7331 \newcommand*\MFUnocap}[1]{\listadd{\@mfu@nocaplist}{#1}}
```

`\gMFUnocap` Global version.

```
7332 \newcommand*\gMFUnocap}[1]{\listgadd{\@mfu@nocaplist}{#1}}
```

`\MFUclear` Clear the list

```
7333 \newcommand*\MFUclear{\renewcommand*\@mfu@nocaplist{}}
```

`\xcapitalisewords` Short-cut command:

```
7334 \newcommand*\xcapitalisewords[1]{%  
7335   \expandafter\capitalisewords\expandafter{#1}%  
7336 }
```

4 Mfirstuc-english Documented Code

```
7337 \NeedsTeXFormat{LaTeX2e}  
7338 \ProvidesPackage{mfirstuc-english}[2014/07/30 v1.0 (NLCT)]
```

Load mfirstuc if not already loaded:

```
7339 \RequirePackage{mfirstuc}
```

Add no-cap words. (List isn't a complete list.)

```
7340 \MFUnocap{a}  
7341 \MFUnocap{an}  
7342 \MFUnocap{and}  
7343 \MFUnocap{but}  
7344 \MFUnocap{for}  
7345 \MFUnocap{in}  
7346 \MFUnocap{of}  
7347 \MFUnocap{or}  
7348 \MFUnocap{no}  
7349 \MFUnocap{nor}  
7350 \MFUnocap{so}  
7351 \MFUnocap{some}  
7352 \MFUnocap{the}  
7353 \MFUnocap{with}  
7354 \MFUnocap{yet}
```

5 Glossary Styles

5.1 Glossary hyper-navigation definitions (glossary-hypernav package)

Package Definition:

```
7355 \ProvidesPackage{glossary-hypernav}[2013/11/14 v4.0 (NLCT)]
```

The commands defined in this package are provided to help navigate around the groups within a glossary (see [subsection 1.15.](#)) `\printglossary` (and `\printglossaries`) set `\@glo@type` to the label of the current glossary. This is used to create a unique hypertextarget in the event of multiple glossaries.

```
\glsnavhyperlink[<type>]{<label>}{<text>}
```

This command makes `<text>` a hyperlink to the glossary group whose label is given by `<label>` for the glossary given by `<type>`.

`\glsnavhyperlink`

```
7356 \newcommand*{\glsnavhyperlink}[3][\@glo@type]{%
7357   \edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%
7358   \@glslink{glsn:#1@#2}{#3}}
```

`\glsnavhypertarget` [*type*] {*label*}{*text*}

This command makes *text* a hypertarget for the glossary group whose label is given by *label* in the glossary given by *type*. If *type* is omitted, `\@glo@type` is used which is set by `\printglossary` to the current glossary label.

`\glsnavhypertarget`

```
7359 \newcommand*{\glsnavhypertarget}[3][\@glo@type]{%
  Add this group to the aux file for re-run check.
7360   \protected@write\@auxout{}{\string\@gls@hypergroup{#1}{#2}}%
  Add the target.
7361   \@glstarget{glsn:#1@#2}{#3}%
  Check list of know groups to determine if a re-run is required.
7362   \expandafter\let
7363     \expandafter\@gls@list\csname @gls@hypergroup@list@#1\endcsname
  Iterate through list and terminate loop if this group is found.
7364   \@for\@gls@elem:=\@gls@list\do{%
7365     \ifthenelse{\equal{\@gls@elem}{#2}}{\@endfortrue}{}%
  Check if list terminated prematurely.
7366   \if@endfor
7367   \else
  This group was not included in the list, so issue a warning.
7368   \GlossariesWarningNoLine{Navigation panel
7369     for glossary type ‘#1’^^Jmissing group ‘#2’}%
7370   \gdef\gls@hypergroup@rerun{%
7371     \GlossariesWarningNoLine{Navigation panel
7372       has changed. Rerun LaTeX}}%
7373   \fi
7374 }
```

`\gls@hypergroup@rerun` Give a warning at the end if re-run required

```
7375 \let\gls@hypergroup@rerun\relax
7376 \AtEndDocument{\gls@hypergroup@rerun}
```

`\@gls@hypergroup` This adds to (or creates) the command `\@gls@hypergroup@list@<glossary type>` which lists all groups for a given glossary, so that the navigation bar only contains those groups that are present. However it requires at least 2 runs to ensure the information is up-to-date.

```
7377 \newcommand*{\@gls@hypergroup}[2]{%
```

```

7378 \@ifundefined{@gls@hypergrouplist@#1}{%
7379   \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{#2}%
7380 }{%
7381   \expandafter\let\expandafter@\gls@tmp
7382     \csname @gls@hypergrouplist@#1\endcsname
7383   \expandafter\xdef\csname @gls@hypergrouplist@#1\endcsname{%
7384     \@gls@tmp,#2}%
7385 }%
7386 }

```

The `\glsnavigation` command displays a simple glossary group navigation. The symbol and number elements are defined separately, so that they can be suppressed if need be. Note that this command will produce a link to all 28 groups, but some groups may not be defined if there are groups that do not contain any terms, in which case you will get an undefined hyperlink warning. Now for the whole navigation bit:

`\glsnavigation`

```

7387 \newcommand*{\glsnavigation}{%
7388 \def@gls@between{}%
7389 \@ifundefined{@gls@hypergrouplist@}@glo@type}{%
7390   \def@gls@list{}%
7391 }{%
7392   \expandafter\let\expandafter@gls@list
7393     \csname @gls@hypergrouplist@}@glo@type\endcsname
7394 }%
7395 \@for@gls@tmp:=@gls@list\do{%
7396   \@gls@between

7397   \@gls@getgrouptitle{@gls@tmp}{@gls@grptitle}%
7398   \glsnavhyperlink{@gls@tmp}{@gls@grptitle}%
7399   \let@gls@between\glshypernavsep%
7400 }%
7401 }

```

`\glshypernavsep` Separator for the hyper navigation bar.

```

7402 \newcommand*{\glshypernavsep}{\space\textbar\space}

```

The `\glssymbolnav` produces a simple navigation set of links for just the symbol and number groups. This used to be used at the start of `\glsnavigation`. This command is no longer needed.

`\glssymbolnav`

```

7403 \newcommand*{\glssymbolnav}{%
7404 \glsnavhyperlink{glssymbols}{\glsgetgrouptitle{glssymbols}}%
7405 \glshypernavsep
7406 \glsnavhyperlink{glsnumbers}{\glsgetgrouptitle{glsnumbers}}%
7407 \glshypernavsep
7408 }

```


5.2 In-line Style (glossary-inline.sty)

This defines an in-line style where the entries are comma-separated with just the name and description displayed.

```
7409 \ProvidesPackage{glossary-inline}[2013/11/14 v4.0 (NLCT)]
```

`inline` Define the inline style.

```
7410 \newglossarystyle{inline}{%
```

Start of glossary sets up first empty separator between entries. (This is then changed by `\glossentry`)

```
7411 \renewenvironment{theglossary}{%
```

```
7412 {%
```

```
7413 \def\gls@inlinesep{}%
```

```
7414 \def\gls@inlinesubsep{}%
```

```
7415 \def\gls@inlinepostchild{}%
```

```
7416 }%
```

```
7417 {\glspostinline}%
```

No header:

```
7418 \renewcommand*{\glossaryheader}{}%
```

No group headings (if heading is required, add `\glsinlinedopostchild` to start definition in case heading follows a child entry):

```
7419 \renewcommand*{\glsgroupheading}[1]{}%
```

Just display separator followed by name and description:

```
7420 \renewcommand{\glossentry}[2]{%
```

```
7421 \glsinlinedopostchild
```

```
7422 \gls@inlinesep
```

```
7423 \glsentryitem{##1}%
```

```
7424 \glsinlinenameformat{##1}{%
```

```
7425 \glossentryname{##1}%
```

```
7426 }%
```

```
7427 \ifglshasdescsuppressed{##1}%
```

```
7428 {%
```

```
7429 \glsinlineemptydescformat
```

```
7430 {%
```

```
7431 \glossentrysymbol{##1}%
```

```
7432 }%
```

```
7433 {%
```

```
7434 ##2%
```

```
7435 }%
```

```
7436 }%
```

```
7437 {%
```

```
7438 \ifglshasdesc{##1}%
```

```
7439 {\glsinlinedescformat{\glossentrydesc{##1}}{\glossentrysymbol{##1}}{##2}}%
```

```
7440 {\glsinlineemptydescformat{\glossentrysymbol{##1}}{##2}}%
```

```
7441 }%
```

```
7442 \ifglshaschildren{##1}%
```

```

7443   {%
7444     \glsresetsubentrycounter
7445     \glsinlineparentchildseparator
7446     \def\gls@inlinesubsep{}%
7447     \def\gls@inlinepostchild{\glsinlinepostchild}%
7448   }%
7449   {}%
7450   \def\gls@inlinesep{\glsinlineseparator}%
7451 }%

```

Sub-entries display description:

```

7452 \renewcommand{\subglossentry}[3]{%
7453   \gls@inlinesubsep%
7454   \glsinlinesubnameformat{##2}%
7455   \glossentryname{##2}%
7456   \glssubentryitem{##2}%
7457   \glsinlinesubdescformat{\glossentrydesc{##2}}{\glossentrysymbol{##2}}{##3}%
7458   \def\gls@inlinesubsep{\glsinlinesubseparator}%
7459 }%

```

Nothing special between groups:

```

7460 \renewcommand*{\glsgroupskip}{}%
7461 }

```

`\glsinlinedopostchild`

```

7462 \newcommand*{\glsinlinedopostchild}{%
7463   \gls@inlinepostchild
7464   \def\gls@inlinepostchild{}%
7465 }

```

`\glsinlineseparator` Separator to use between entries.

```

7466 \newcommand*{\glsinlineseparator}{;\space}

```

`\glsinlinesubseparator` Separator to use between sub-entries.

```

7467 \newcommand*{\glsinlinesubseparator}{,\space}

```

`\glsinlineparentchildseparator` Separator to use between parent and children.

```

7468 \newcommand*{\glsinlineparentchildseparator}{:\space}

```

`\glsinlinepostchild` Hook to use between child and next entry

```

7469 \newcommand*{\glsinlinepostchild}{}

```

`\glspostinline` Terminator for inline glossary.

```

7470 \newcommand*{\glspostinline}{\glspostdescription\space}

```

`\glsinlinenameformat` Formats the name of the entry (first argument label, second argument name):

```

7471 \newcommand*{\glsinlinenameformat}[2]{\glstarget{#1}{#2}}

```

`\glsinlinedescformat` Formats the entry's description, symbol and location list:

```

7472 \newcommand*{\glsinlinedescformat}[3]{\space#1}

```

`inlineemptydescformat` Formats the entry's symbol and location list when the description is empty:

```
7473 \newcommand*\glsinlineemptydescformat}[2]{}
```

`inlinesubnameformat` Formats the name of the subentry (first argument label, second argument name):

```
7474 \newcommand*\glsinlinesubnameformat}[2]{\glstarget{#1}{}}
```

`inlinesubdescformat` Formats the subentry's description, symbol and location list:

```
7475 \newcommand*\glsinlinesubdescformat}[3]{#1}
```

5.3 List Style (glossary-list.sty)

The style file defines glossary styles that use the description environment. Note that since the entry name is placed in the optional argument to the `\item` command, it will appear in a bold font by default.

```
7476 \ProvidesPackage{glossary-list}[2013/11/14 v4.0 (NLCT)]
```

`list` The list glossary style uses the description environment. The group separator `\glsgroupskip` is redefined as `\indexspace` which produces a gap between groups. The glossary heading and the group headings do nothing. Sub-entries immediately follow the main entry without the sub-entry name. This style does not use the entry's symbol. This is used as the default style for the glossaries package.

```
7477 \newglossarystyle{list}{%
```

Use description environment:

```
7478 \renewenvironment{theglossary}{%
```

```
7479 \begin{description}}{\end{description}}%
```

No header at the start of the environment:

```
7480 \renewcommand*\glossaryheader}{}
```

No group headings:

```
7481 \renewcommand*\glsgroupheading}[1]{}
```

Main (level 0) entries start a new item in the list:

```
7482 \renewcommand*\glossentry}[2]{%
```

```
7483 \item[\glsentryitem{##1}]%
```

```
7484 \glstarget{##1}{\glossentryname{##1}}]
```

```
7485 \glossentrydesc{##1}\glspostdescription\space ##2}]%
```

Sub-entries continue on the same line:

```
7486 \renewcommand*\subglossentry}[3]{%
```

```
7487 \glssubentryitem{##2}]%
```

```
7488 \glstarget{##2}{\strut}]%
```

```
7489 \glossentrydesc{##2}\glspostdescription\space ##3.}]%
```

```
7490 % \end{macrocode}
```

```
7491 % Add vertical space between groups:
```

```
7492 %\changes{3.03}{2012/09/21}{added check for glsnogroupskip}
```

```

7493 % \begin{macrocode}
7494 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
7495 }

```

`listgroup` The `listgroup` style is like the `list` style, but the glossary groups have headings.

```

7496 \newglossarystyle{listgroup}{%
  Base it on the list style:
7497 \setglossarystyle{list}%
  Each group has a heading:
7498 \renewcommand*{\glsgroupheading}[1]{\item[\glsgetgrouptitle{##1}]}

```

`listhypergroup` The `listhypergroup` style is like the `listgroup` style, but has a set of links to the groups at the start of the glossary.

```

7499 \newglossarystyle{listhypergroup}{%
  Base it on the list style:
7500 \setglossarystyle{list}%
  Add navigation links at the start of the environment:
7501 \renewcommand*{\glossaryheader}{%
7502 \item[\glsnavigation]}%
  Each group has a heading with a hypertext:
7503 \renewcommand*{\glsgroupheading}[1]{%
7504 \item[\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}]}

```

`altlist` The `altlist` glossary style is like the `list` style, but places the description on a new line. Sub-entries follow in separate paragraphs without the sub-entry name. This style does not use the entry's symbol.

```

7505 \newglossarystyle{altlist}{%
  Base it on the list style:
7506 \setglossarystyle{list}%
  Main (level 0) entries start a new item in the list with a line break after the entry name:
7507 \renewcommand*{\glossentry}[2]{%
7508 \item[\glsentryitem{##1}%
7509 \glstarget{##1}{\glossentryname{##1}}]%
  Version 3.04 changed \newline to the following paragraph break stuff (thanks to Daniel Gebhardt for supplying the fix) to prevent a page break occurring at this point.
7510 \mbox{}\par\nobreak\@afterheading
7511 \glossentrydesc{##1}\glspostdescription\space ##2}%
  Sub-entries start a new paragraph:
7512 \renewcommand{\subglossentry}[3]{%
7513 \par
7514 \glssubentryitem{##2}%

```

```

7515   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space ##3}%
7516 }

```

`altlistgroup` The `altlistgroup` glossary style is like the `altlist` style, but the glossary groups have headings.

```

7517 \newglossarystyle{altlistgroup}{%
      Base it on the altlist style:
7518   \setglossarystyle{altlist}%
      Each group has a heading:
7519   \renewcommand*{\glsgroupheading}[1]{\item[\glsgetgrouptitle{##1}]}

```

`altlisthypergroup` The `altlisthypergroup` glossary style is like the `altlistgroup` style, but has a set of links to the groups at the start of the glossary.

```

7520 \newglossarystyle{altlisthypergroup}{%
      Base it on the altlist style:
7521   \setglossarystyle{altlist}%
      Add navigation links at the start of the environment:
7522   \renewcommand*{\glossaryheader}{%
7523     \item[\glsnavigation]}%
      Each group has a heading with a hypertext:
7524   \renewcommand*{\glsgroupheading}[1]{%
7525     \item[\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}]}

```

`listdotted` The `listdotted` glossary style was supplied by Axel Menzel. I've modified it slightly so that the distance from the start of the name to the end of the dotted line is specified by `\glslistdottedwidth`. Note that this style ignores the page numbers as well as the symbol. Sub-entries are displayed in the same way as top-level entries.

```

7526 \newglossarystyle{listdotted}{%
      Base it on the list style:
7527   \setglossarystyle{list}%
      Each main (level 0) entry starts a new item:
7528   \renewcommand*{\glossentry}[2]{%
7529     \item[]\makebox[\glslistdottedwidth][1]{%
7530       \glstarget{##1}{\glsentryitem{##1}}%
7531       \glstarget{##1}{\glossentryname{##1}}%
7532       \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##1}}%
      Sub entries have the same format as main entries:
7533   \renewcommand*{\subglossentry}[3]{%
7534     \item[]\makebox[\glslistdottedwidth][1]{%
7535       \glssubentryitem{##2}}%
7536     \glstarget{##2}{\glossentryname{##2}}%
7537     \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}\glossentrydesc{##2}}%
7538 }

```

`\glslistdottedwidth`

```
7539 \newlength\glslistdottedwidth
7540 \setlength{\glslistdottedwidth}{.5\hsize}
```

`sublistdotted` This style is similar to the `glostylelistdotted` style, except that the main entries just have the name displayed.

```
7541 \newglossarystyle{sublistdotted}{%
  Base it on the listdotted style:
7542 \setglossarystyle{listdotted}%
  Main (level 0) entries just display the name:
7543 \renewcommand*{\glossentry}[2]{%
7544 \item[\glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}}]}%
7545 }
```

5.4 Glossary Styles using `longtable` (the `glossary-long` package)

The glossary styles defined in the package used the `longtable` environment in the glossary.

```
7546 \ProvidesPackage{glossary-long}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7547 \RequirePackage{longtable}
```

`\glsdescwidth` This is a length that governs the width of the description column. (There's a chance that the user may specify `nolong` and then load later, in which case `\glsdescwidth` may have already been defined by `.` The same goes for `\glspagelistwidth`.)

```
7548 \@ifundefined{glsdescwidth}{%
7549 \newlength\glsdescwidth
7550 \setlength{\glsdescwidth}{0.6\hsize}
7551 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column.

```
7552 \@ifundefined{glspagelistwidth}{%
7553 \newlength\glspagelistwidth
7554 \setlength{\glspagelistwidth}{0.1\hsize}
7555 }{}
```

`long` The long glossary style command which uses the `longtable` environment:

```
7556 \newglossarystyle{long}{%
  Use longtable with two columns:
7557 \renewenvironment{theglossary}%
7558 {\begin{longtable}\lp{\glsdescwidth}}%
7559 {\end{longtable}}%
  Do nothing at the start of the environment:
7560 \renewcommand*{\glossaryheader}{}%
```

No heading between groups:

```
7561 \renewcommand*\glsgroupheading}[1]{%
```

Main (level 0) entries displayed in a row:

```
7562 \renewcommand{\glossentry}[2]{%
7563   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7564   \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
7565 }%
```

Sub entries displayed on the following row without the name:

```
7566 \renewcommand{\subglossentry}[3]{%
7567   &
7568   \glssubentryitem{##2}%
7569   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
7570   ##3\tabularnewline
7571 }%
```

Blank row between groups:

```
7572 \renewcommand*\glsgroupskip{\ifglsnogroupskip\else &
7573 \tabularnewline\fi}%
7574 }
```

longborder The longborder style is like the above, but with horizontal and vertical lines:

```
7575 \newglossarystyle{longborder}{%
```

Base it on the glostylelong style:

```
7576 \setglossarystyle{long}%
```

Use longtable with two columns with vertical lines between each column:

```
7577 \renewenvironment{theglossary}{%
7578   \begin{longtable}[|l|p{\glsdescwidth}|]{\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
7579 \renewcommand*\glossaryheader{\hline\endhead\hline\endfoot}%
7580 }
```

longheader The longheader style is like the long style but with a header:

```
7581 \newglossarystyle{longheader}{%
```

Base it on the glostylelong style:

```
7582 \setglossarystyle{long}%
```

Set the table's header:

```
7583 \renewcommand*\glossaryheader{%
7584   \bfseries \entryname & \bfseries \descriptionname\tabularnewline\endhead}%
7585 }
```

longheaderborder The longheaderborder style is like the long style but with a header and border:

```
7586 \newglossarystyle{longheaderborder}{%
```

Base it on the glostylelongborder style:

```
7587 \setglossarystyle{longborder}%
```

Set the table's header and add horizontal line to table's foot:

```
7588 \renewcommand*{\glossaryheader}{%
7589 \hline\bfseries \entryname & \bfseries
7590 \descriptionname\tabularnewline\hline
7591 \endhead
7592 \hline\endfoot}%
7593 }
```

`long3col` The `long3col` style is like `long` but with 3 columns

```
7594 \newglossarystyle{long3col}{%
```

Use a `longtable` with 3 columns:

```
7595 \renewenvironment{theglossary}%
7596 {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
7597 {\end{longtable}}%
```

No table header:

```
7598 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
7599 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
7600 \renewcommand{\glossentry}[2]{%
7601 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7602 \glossentrydesc{##1} & ##2\tabularnewline
7603 }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
7604 \renewcommand{\subglossentry}[3]{%
7605 &
7606 \glssubentryitem{##2}%
7607 \glstarget{##2}{\strut}\glossentrydesc{##2} &
7608 ##3\tabularnewline
7609 }%
```

Blank row between groups:

```
7610 \renewcommand*{\glsgroupskip}{%
7611 \ifglsnogroupskip\else & &\tabularnewline\fi}%
7612 }
```

`long3colborder` The `long3colborder` style is like the `long3col` style but with a border:

```
7613 \newglossarystyle{long3colborder}{%
```

Base it on the `glostylelong3col` style:

```
7614 \setglossarystyle{long3col}%
```

Use a `longtable` with 3 columns with vertical lines around them:

```
7615 \renewenvironment{theglossary}%
7616 {\begin{longtable}{|l|p{\glsdescwidth}|p{\glspagelistwidth}|}%
7617 {\end{longtable}}%
```


Place horizontal lines at the head and foot of the table:

```
7618 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7619 }
```

`long3colheader` The `long3colheader` style is like `long3col` but with a header row:

```
7620 \newglossarystyle{long3colheader}{%
Base it on the glostylelong3col style:
7621 \setglossarystyle{long3col}%
Set the table's header:
7622 \renewcommand*{\glossaryheader}{%
7623 \bfseries\entryname&\bfseries\descriptionname&
7624 \bfseries\pagelistname\tabularnewline\endhead}%
7625 }
```

`long3colheaderborder` The `long3colheaderborder` style is like the above but with a border

```
7626 \newglossarystyle{long3colheaderborder}{%
Base it on the glostylelong3colborder style:
7627 \setglossarystyle{long3colborder}%
Set the table's header and add horizontal line at table's foot:
7628 \renewcommand*{\glossaryheader}{%
7629 \hline
7630 \bfseries\entryname&\bfseries\descriptionname&
7631 \bfseries\pagelistname\tabularnewline\hline\endhead
7632 \hline\endfoot}%
7633 }
```

`long4col` The `long4col` style has four columns where the third column contains the value of the associated symbol key.

```
7634 \newglossarystyle{long4col}{%
Use a longtable with 4 columns:
7635 \renewenvironment{theglossary}%
7636 {\begin{longtable}{llll}}%
7637 {\end{longtable}}%
No table header:
7638 \renewcommand*{\glossaryheader}{}%
No group headings:
7639 \renewcommand*{\glsgroupheading}[1]{}%
Main (level 0) entries on a single row (name in first column, description in second column, symbol in third column, page list in last column):
7640 \renewcommand{\glossentry}[2]{%
7641 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7642 \glossentrydesc{##1} &
7643 \glossentrysymbol{##1} &
7644 ##2\tabularnewline
7645 }%
```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```
7646 \renewcommand{\subglossentry}[3]{%
7647     &
7648     \glssubentryitem{##2}%
7649     \glstarget{##2}{\strut}\glossentrydesc{##2} &
7650     \glossentrysymbol{##2} & ##3\tabularnewline
7651 }%
```

Blank row between groups:

```
7652 \renewcommand*{\glsgroupskip}{%
7653     \ifglsnogroupskip\else & & \tabularnewline\fi}%
7654 }
```

`long4colheader` The `long4colheader` style is like `long4col` but with a header row.

```
7655 \newglossarystyle{long4colheader}{%
    Base it on the glostylelong4col style:
7656     \setglossarystyle{long4col}%
    Table has a header:
7657     \renewcommand*{\glossaryheader}{%
7658         \bfseries\entryname&\bfseries\descriptionname&
7659         \bfseries \symbolname&
7660         \bfseries\pagelistname\tabularnewline\endhead}%
7661 }
```

`long4colborder` The `long4colborder` style is like `long4col` but with a border.

```
7662 \newglossarystyle{long4colborder}{%
    Base it on the glostylelong4col style:
7663     \setglossarystyle{long4col}%
    Use a longtable with 4 columns surrounded by vertical lines:
7664     \renewenvironment{theglossary}%
7665         {\begin{longtable}{|l|l|l|l|}}%
7666         {\end{longtable}}%
    Add horizontal lines to the head and foot of the table:
7667     \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7668 }
```

`long4colheaderborder` The `long4colheaderborder` style is like the above but with a border.

```
7669 \newglossarystyle{long4colheaderborder}{%
    Base it on the glostylelong4col style:
7670     \setglossarystyle{long4col}%
    Use a longtable with 4 columns surrounded by vertical lines:
7671     \renewenvironment{theglossary}%
7672         {\begin{longtable}{|l|l|l|l|}}%
7673         {\end{longtable}}%
```

Add table header and horizontal line at the table's foot:

```
7674 \renewcommand*{\glossaryheader}{%
7675 \hline\bfseries\entryname&\bfseries\descriptionname&
7676 \bfseries \symbolname&
7677 \bfseries\pagelistname\tabularnewline\hline\endhead
7678 \hline\endfoot}%
7679 }
```

`altlong4col` The `altlong4col` style is like the `long4col` style but can have multiline descriptions and page lists.

```
7680 \newglossarystyle{altlong4col}{%
```

Base it on the `glostylelong4col` style:

```
7681 \setglossarystyle{long4col}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7682 \renewenvironment{theglossary}%
7683 {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}%
7684 {\end{longtable}}%
7685 }
```

`altlong4colheader` The `altlong4colheader` style is like `altlong4col` but with a header row.

```
7686 \newglossarystyle{altlong4colheader}{%
```

Base it on the `glostylelong4colheader` style:

```
7687 \setglossarystyle{long4colheader}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7688 \renewenvironment{theglossary}%
7689 {\begin{longtable}{lp{\glsdescwidth}lp{\glspagelistwidth}}%
7690 {\end{longtable}}%
7691 }
```

`altlong4colborder` The `altlong4colborder` style is like `altlong4col` but with a border.

```
7692 \newglossarystyle{altlong4colborder}{%
```

Base it on the `glostylelong4colborder` style:

```
7693 \setglossarystyle{long4colborder}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7694 \renewenvironment{theglossary}%
7695 {\begin{longtable}{|lp{\glsdescwidth}|lp{\glspagelistwidth}|}%
7696 {\end{longtable}}%
7697 }
```

`altlong4colheaderborder` The `altlong4colheaderborder` style is like the above but with a header as well as a border.

```
7698 \newglossarystyle{altlong4colheaderborder}{%
```

Base it on the `glostylelong4colheaderborder` style:

```
7699 \setglossarystyle{long4colheaderborder}%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7700 \renewenvironment{theglossary}%  
7701   {\begin{longtable}{|l|p{\glsdescwidth}|l|p{\glspagelistwidth}|}%  
7702   {\end{longtable}}%  
7703 }
```

5.5 Glossary Styles using `longtable` (the `glossary-longragged` package)

The glossary styles defined in the package used the `longtable` environment in the glossary and use ragged right formatting for the multiline columns.

```
7704 \ProvidesPackage{glossary-longragged}[2014/07/30 v4.08 (NLCT)]
```

Requires the package:

```
7705 \RequirePackage{array}
```

Requires the package:

```
7706 \RequirePackage{longtable}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may have already been defined.

```
7707 \@ifundefined{glsdescwidth}{%  
7708   \newlength\glsdescwidth  
7709   \setlength{\glsdescwidth}{0.6\hsize}  
7710 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
7711 \@ifundefined{glspagelistwidth}{%  
7712   \newlength\glspagelistwidth  
7713   \setlength{\glspagelistwidth}{0.1\hsize}  
7714 }{}
```

`longragged` The `longragged` glossary style is like the `long` but uses ragged right formatting for the description column.

```
7715 \newglossarystyle{longragged}{%
```

Use `longtable` with two columns:

```
7716 \renewenvironment{theglossary}%  
7717   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}}}%  
7718   {\end{longtable}}%
```

Do nothing at the start of the environment:

```
7719 \renewcommand*\glossaryheader{}%
```

No heading between groups:

```
7720 \renewcommand*\glsgroupheading}[1]{%
```

Main (level 0) entries displayed in a row:

```
7721 \renewcommand{\glossentry}[2]{%
7722   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7723   \glossentrydesc{##1}\glspostdescription\space ##2%
7724   \tabularnewline
7725 }%
```

Sub entries displayed on the following row without the name:

```
7726 \renewcommand{\subglossentry}[3]{%
7727   &
7728   \glssubentryitem{##2}%
7729   \glstarget{##2}{\strut}\glossentrydesc{##2}%
7730   \glspostdescription\space ##3%
7731   \tabularnewline
7732 }%
```

Blank row between groups:

```
7733 \renewcommand*\glsgroupskip{\ifglsnogroupskip\else & \tabularnewline\fi}%
7734 }
```

`longraggedborder` The `longraggedborder` style is like the above, but with horizontal and vertical lines:

```
7735 \newglossarystyle{longraggedborder}{%
```

Base it on the `glostylelongragged` style:

```
7736 \setglossarystyle{longragged}{%
```

Use `longtable` with two columns with vertical lines between each column:

```
7737 \renewenvironment{theglossary}{%
7738   \begin{longtable}[1|>{\raggedright}p{\glsdescwidth}|]}%
7739   {\end{longtable}}%
```

Place horizontal lines at the head and foot of the table:

```
7740 \renewcommand*\glossaryheader{\hline\endhead\hline\endfoot}%
7741 }
```

`longraggedheader` The `longraggedheader` style is like the `longragged` style but with a header:

```
7742 \newglossarystyle{longraggedheader}{%
```

Base it on the `glostylelongragged` style:

```
7743 \setglossarystyle{longragged}{%
```

Set the table's header:

```
7744 \renewcommand*\glossaryheader{%
7745   \bfseries \entryname & \bfseries \descriptionname
7746   \tabularnewline\endhead}%
7747 }
```

`longraggedheaderborder` The `longraggedheaderborder` style is like the `longragged` style but with a header and border:

```
7748 \newglossarystyle{longraggedheaderborder}{%
```

Base it on the `glostylelongraggedborder` style:

```
7749 \setglossarystyle{longraggedborder}{%
```

Set the table's header and add horizontal line to table's foot:

```
7750 \renewcommand*{\glossaryheader}{%
7751 \hline\bfseries \entryname & \bfseries \descriptionname
7752 \tabularnewline\hline
7753 \endhead
7754 \hline\endfoot}%
7755 }
```

`longragged3col` The `longragged3col` style is like `longragged` but with 3 columns

```
7756 \newglossarystyle{longragged3col}{%
```

Use a `longtable` with 3 columns:

```
7757 \renewenvironment{theglossary}{%
7758 \begin{longtable}[1>{\raggedright}p{\glsdescwidth}%
7759 >{\raggedright}p{\glspagelistwidth}}{%
7760 \end{longtable}}%
```

No table header:

```
7761 \renewcommand*{\glossaryheader}{}%
```

No headings between groups:

```
7762 \renewcommand*{\glsgroupheading}[1]{%}
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
7763 \renewcommand{\glossentry}[2]{%
7764 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7765 \glossentrydesc{##1} & ##2\tabularnewline
7766 }%
```

Sub-entries on a separate row (no name, description in second column, page list in third column):

```
7767 \renewcommand{\subglossentry}[3]{%
7768 &
7769 \glssubentryitem{##2}%
7770 \glstarget{##2}{\strut}\glossentrydesc{##2} &
7771 ##3\tabularnewline
7772 }%
```

Blank row between groups:

```
7773 \renewcommand*{\glsgroupskip}{%
7774 \ifglsnogroupskip\else & &\tabularnewline\fi}%
7775 }
```

`longragged3colborder` The `longragged3colborder` style is like the `longragged3col` style but with a border:

```
7776 \newglossarystyle{longragged3colborder}{%
```

Base it on the `glostylelongragged3col` style:

```
7777 \setglossarystyle{longragged3col}{%
```

Use a `longtable` with 3 columns with vertical lines around them:

```
7778 \renewenvironment{theglossary}{%
```

```
7779   {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}}|%
```

```
7780     >{\raggedright}p{\glspagelistwidth}}|}%
```

```
7781   {\end{longtable}}}%
```

Place horizontal lines at the head and foot of the table:

```
7782 \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
```

```
7783 }
```

`longragged3colheader` The `longragged3colheader` style is like `longragged3col` but with a header row:

```
7784 \newglossarystyle{longragged3colheader}{%
```

Base it on the `glostylelongragged3col` style:

```
7785 \setglossarystyle{longragged3col}{%
```

Set the table's header:

```
7786 \renewcommand*{\glossaryheader}{%
```

```
7787   \bfseries\entryname&\bfseries\descriptionname&
```

```
7788   \bfseries\pagelistname\tabularnewline\endhead}%
```

```
7789 }
```

`longragged3colheaderborder` The `longragged3colheaderborder` style is like the above but with a border

```
7790 \newglossarystyle{longragged3colheaderborder}{%
```

Base it on the `glostylelongragged3colborder` style:

```
7791 \setglossarystyle{longragged3colborder}{%
```

Set the table's header and add horizontal line at table's foot:

```
7792 \renewcommand*{\glossaryheader}{%
```

```
7793   \hline
```

```
7794   \bfseries\entryname&\bfseries\descriptionname&
```

```
7795   \bfseries\pagelistname\tabularnewline\hline\endhead
```

```
7796   \hline\endfoot}%
```

```
7797 }
```

`altlongragged4col` The `altlongragged4col` style is like the `altlong4col` style defined in the package, except that ragged right formatting is used for the description and page list columns.

```
7798 \newglossarystyle{altlongragged4col}{%
```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```
7799 \renewenvironment{theglossary}{%
```

```

7800   {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
7801     >{\raggedright}p{\glspagelistwidth}}}%
7802   {\end{longtable}}%

```

No table header:

```
7803   \renewcommand*{\glossaryheader}{}%

```

No group headings:

```
7804   \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries on a single row (name in first column, description in second column, symbol in third column, page list in last column):

```

7805   \renewcommand{\glossentry}[2]{%
7806     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7807     \glossentrydesc{##1} & \glossentrysymbol{##1} &
7808     ##2\tabularnewline
7809   }%

```

Sub entries on a single row with no name (description in second column, symbol in third column, page list in last column):

```

7810   \renewcommand{\subglossentry}[3]{%
7811     &
7812     \glsesubentryitem{##2}%
7813     \glstarget{##2}{\strut}\glossentrydesc{##2} &
7814     \glossentrysymbol{##2} & ##3\tabularnewline
7815   }%

```

Blank row between groups:

```

7816   \renewcommand*{\glsgroupskip}{%
7817     \ifglsnogroupskip\else & & \tabularnewline\fi}%
7818 }

```

`ongragged4colheader` The `altlongragged4colheader` style is like `altlongragged4col` but with a header row.

```
7819 \newglossarystyle{altlongragged4colheader}{%

```

Base it on the `glostylealtlongragged4col` style:

```
7820   \setglossarystyle{altlongragged4col}%

```

Use a `longtable` with 4 columns where the second and last columns may have multiple lines in each row:

```

7821   \renewenvironment{theglossary}%
7822     {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
7823       >{\raggedright}p{\glspagelistwidth}}}%
7824     {\end{longtable}}%

```

Table has a header:

```

7825   \renewcommand*{\glossaryheader}{%
7826     \bfseries\entryname&\bfseries\descriptionname&
7827     \bfseries \symbolname&
7828     \bfseries\pagelistname\tabularnewline\endhead}%
7829 }

```


`altragged4colborder` The `altragged4colborder` style is like `altragged4col` but with a border.

```

7830 \newglossarystyle{altragged4colborder}{%
      Base it on the glostylealtragged4col style:
7831   \setglossarystyle{altragged4col}%

      Use a longtable with 4 columns where the second and last columns may have
      multiple lines in each row:
7832   \renewenvironment{theglossary}%
7833     {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|l|}%
7834       >{\raggedright}p{\glspagelistwidth}|}%
7835     {\end{longtable}}%

      Add horizontal lines to the head and foot of the table:
7836   \renewcommand*{\glossaryheader}{\hline\endhead\hline\endfoot}%
7837 }
```

`altragged4colheaderborder` The `altragged4colheaderborder` style is like the above but with a header as well as a border.

```

7838 \newglossarystyle{altragged4colheaderborder}{%
      Base it on the glostylealtragged4col style:
7839   \setglossarystyle{altragged4col}%

      Use a longtable with 4 columns where the second and last columns may have
      multiple lines in each row:
7840   \renewenvironment{theglossary}%
7841     {\begin{longtable}{|l|>{\raggedright}p{\glsdescwidth}|l|}%
7842       >{\raggedright}p{\glspagelistwidth}|}%
7843     {\end{longtable}}%

      Add table header and horizontal line at the table's foot:
7844   \renewcommand*{\glossaryheader}{%
7845     \hline\bfseries\entryname&\bfseries\descriptionname&
7846     \bfseries \symbolname&
7847     \bfseries\pagelistname\tabularnewline\hline\endhead
7848     \hline\endfoot}%
7849 }
```

5.6 Glossary Styles using multicol (glossary-mcols.sty)

The style file defines glossary styles that use the `multicol` package. These use the tree-like glossary styles in a `multicol` environment.

```

7850 \ProvidesPackage{glossary-mcols}[2013/11/14 v4.0 (NLCT)]
```

Required packages:

```

7851 \RequirePackage{multicol}
7852 \RequirePackage{glossary-tree}
```

`\glscols` Define macro in which to store the number of columns. (Defaults to 2.)

```

7853 \newcommand*{\glscols}{2}
```

`mcolindex` Multi-column index style. Same as the `index`, but puts the glossary in multiple columns. (Ideally the glossary title should go in the optional argument of `multicols`, but the title isn't part of the glossary style.)

```

7854 \newglossarystyle{mcolindex}{%
7855   \setglossarystyle{index}%
7856   \renewenvironment{theglossary}%
7857     {%

7858     \begin{multicols}{\glsmcols}
7859     \setlength{\parindent}{0pt}%
7860     \setlength{\parskip}{0pt plus 0.3pt}%
7861     \let\item\@idxitem}%
7862   {\end{multicols}}%
7863 }
```

`mcolindexgroup` As `mcolindex` but has headings:

```

7864 \newglossarystyle{mcolindexgroup}{%
7865   \setglossarystyle{mcolindex}%
7866   \renewcommand*{\glsgroupheading}[1]{%
7867     \item\textbf{\glsgetgrouptitle{##1}}\indexspace}%
7868 }
```

`mcolindexhypergroup` The `mcolindexhypergroup` style is like the `mcolindexgroup` style but has hyper navigation.

```

7869 \newglossarystyle{mcolindexhypergroup}{%

  Base it on the glostylemcolindex style:
7870   \setglossarystyle{mcolindex}%

  Put navigation links to the groups at the start of the glossary:
7871   \renewcommand*{\glossaryheader}{%
7872     \item\textbf{\glsnavigation}\indexspace}%

  Add a heading for each group (with a target). The group's title is in bold followed
  by a vertical gap.
7873   \renewcommand*{\glsgroupheading}[1]{%
7874     \item\textbf{\glsnavhypertarget{##1}}{\glsgetgrouptitle{##1}}}%
7875     \indexspace}%
7876 }
```

`mcoltree` Multi-column index style. Same as the `tree`, but puts the glossary in multiple columns.

```

7877 \newglossarystyle{mcoltree}{%
7878   \setglossarystyle{tree}%
7879   \renewenvironment{theglossary}%
7880     {%

7881     \begin{multicols}{\glsmcols}
7882     \setlength{\parindent}{0pt}%
7883     \setlength{\parskip}{0pt plus 0.3pt}%

```

```

7884 }%
7885 {\end{multicols}}}%
7886 }

```

`mcoltreegroup` Like the `mcoltree` style but the glossary groups have headings.

```

7887 \newglossarystyle{mcoltreegroup}{%
  Base it on the glostylemcoltree style:
7888 \setglossarystyle{mcoltree}%
  Each group has a heading (in bold) followed by a vertical gap):
7889 \renewcommand{\glsgroupheading}[1]{\par
7890 \noindent\textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
7891 }

```

`mcoltreehypergroup` The `mcoltreehypergroup` style is like the `treegroup` style, but has a set of links to the groups at the start of the glossary.

```

7892 \newglossarystyle{mcoltreehypergroup}{%
  Base it on the glostylemcoltree style:
7893 \setglossarystyle{mcoltree}%
  Put navigation links to the groups at the start of the theglossary environment:
7894 \renewcommand*{\glossaryheader}{%
7895 \par\noindent\textbf{\glsnavigation}\par\indexspace}%
  Each group has a heading (in bold with a target) followed by a vertical gap):
7896 \renewcommand*{\glsgroupheading}[1]{%
7897 \par\noindent
7898 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
7899 \indexspace}%
7900 }

```

`mcoltreename` Multi-column index style. Same as the `treename`, but puts the glossary in multiple columns.

```

7901 \newglossarystyle{mcoltreename}{%
7902 \setglossarystyle{treename}%
7903 \renewenvironment{theglossary}%
7904 {%
7905 \begin{multicols}{\glsncols}
7906 \setlength{\parindent}{0pt}%
7907 \setlength{\parskip}{0pt plus 0.3pt}%
7908 }%
7909 {\end{multicols}}}%
7910 }

```

`mcoltreenamegroup` Like the `mcoltreename` style but the glossary groups have headings.

```

7911 \newglossarystyle{mcoltreenamegroup}{%
  Base it on the glostylemcoltreename style:
7912 \setglossarystyle{mcoltreename}%

```

Give each group a heading:

```
7913 \renewcommand{\glsgroupheading}[1]{\par
7914   \noindent\textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
7915 }
```

`reenamehypergroup` The `mcoltreenonamehypergroup` style is like the `mcoltreenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
7916 \newglossarystyle{mcoltreenonamehypergroup}{%
```

Base it on the `glostylemcoltreenoname` style:

```
7917 \setglossarystyle{mcoltreenoname}%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```
7918 \renewcommand*{\glossaryheader}{%
7919   \par\noindent\textbf{\glsnavigation}\par\indexspace}%
```

Each group has a heading (in bold with a target) followed by a vertical gap):

```
7920 \renewcommand*{\glsgroupheading}[1]{%
7921   \par\noindent
7922   \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
7923   \indexspace}%
7924 }
```

`mcolalmtree` Multi-column index style. Same as the `almtree`, but puts the glossary in multiple columns.

```
7925 \newglossarystyle{mcolalmtree}{%
7926   \setglossarystyle{almtree}%
7927   \renewenvironment{theglossary}%
7928   {%
```

```
7929     \begin{multicols}{\glsmcols}
7930     \def\@gls@prevlevel{-1}%
7931     \mbox{}\par
7932   }%
7933   {\par\end{multicols}}%
7934 }
```

`mcolalmtreegroup` Like the `mcolalmtree` style but the glossary groups have headings.

```
7935 \newglossarystyle{mcolalmtreegroup}{%
```

Base it on the `glostylemcolalmtree` style:

```
7936 \setglossarystyle{mcolalmtree}%
```

Give each group a heading.

```
7937 \renewcommand{\glsgroupheading}[1]{\par
7938   \def\@gls@prevlevel{-1}%
7939   \hangindent0pt\relax
7940   \parindent0pt\relax
7941   \textbf{\glsgetgrouptitle{##1}}\par\indexspace}%
7942 }
```

`colalmtreehypergroup` The `colalmtreehypergroup` style is like the `colalmtreegroup` style, but has a set of links to the groups at the start of the glossary.

```
7943 \newglossarystyle{colalmtreehypergroup}{%
```

Base it on the `glostylemcolalmtree` style:

```
7944 \setglossarystyle{colalmtree}{%
```

Put the navigation links in the header

```
7945 \renewcommand*{\glossaryheader}{%
```

```
7946 \par
```

```
7947 \def\@gls@prevlevel{-1}%
```

```
7948 \hangindent0pt\relax
```

```
7949 \parindent0pt\relax
```

```
7950 \textbf{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```
7951 \renewcommand*{\glsgroupheading}[1]{%
```

```
7952 \par
```

```
7953 \def\@gls@prevlevel{-1}%
```

```
7954 \hangindent0pt\relax
```

```
7955 \parindent0pt\relax
```

```
7956 \textbf{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
```

```
7957 \indexspace}}
```

5.7 Glossary Styles using supertabular environment (`glossary-super` package)

The glossary styles defined in the package use the `supertabular` environment.

```
7958 \ProvidesPackage{glossary-super}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
7959 \RequirePackage{supertabular}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined if `glossary-super` has been loaded.

```
7960 \@ifundefined{glsdescwidth}{%
```

```
7961 \newlength\glsdescwidth
```

```
7962 \setlength{\glsdescwidth}{0.6\hsize}
```

```
7963 }{}}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined if `glossary-super` has been loaded.

```
7964 \@ifundefined{glspagelistwidth}{%
```

```
7965 \newlength\glspagelistwidth
```

```
7966 \setlength{\glspagelistwidth}{0.1\hsize}
```

```
7967 }{}}
```

`super` The `super` glossary style uses the `supertabular` environment (it uses lengths defined in the package.)

```
7968 \newglossarystyle{super}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
7969 \renewenvironment{theglossary}%
7970   {\tablehead{}\tabletail{}}%
7971   \begin{supertabular}{lp{\glsdescwidth}}%
7972   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
7973 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
7974 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
7975 \renewcommand{\glossentry}[2]{%
7976   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
7977   \glossentrydesc{##1}\glspostdescription\space ##2\tabularnewline
7978   }%
```

Sub entries put in a row (no name, description and page list in second column):

```
7979 \renewcommand{\subglossentry}[3]{%
7980   &
7981   \glsesubentryitem{##2}%
7982   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
7983   ##3\tabularnewline
7984   }%
```

Blank row between groups:

```
7985 \renewcommand*{\glsgroupskip}{%
7986   \ifglsnogroupskip\else & \tabularnewline\fi}%
7987 }
```

superborder The superborder style is like the above, but with horizontal and vertical lines:

```
7988 \newglossarystyle{superborder}{%
```

Base it on the glostylesuper style:

```
7989 \setglossarystyle{super}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
7990 \renewenvironment{theglossary}%
7991   {\tablehead{\hline}\tabletail{\hline}}%
7992   \begin{supertabular}{|lp{\glsdescwidth}|}%
7993   {\end{supertabular}}%
7994 }
```

superheader The superheader style is like the super style, but with a header:

```
7995 \newglossarystyle{superheader}{%
```

Base it on the `glostylesuper` style:

```
7996 \setglossarystyle{super}%
```

Put the glossary in a `supertabular` environment with two columns, a header and no tail:

```
7997 \renewenvironment{theglossary}%
7998   {\tablehead{\bfseries \entryname &
7999     \bfseries\descriptionname\tabularnewline}%
8000   \tabletail{}}%
8001   \begin{supertabular}{lp{\glsdescwidth}}%
8002   {\end{supertabular}}%
8003 }
```

`superheaderborder` The `superheaderborder` style is like the `super` style but with a header and border:

```
8004 \newglossarystyle{superheaderborder}{%
```

Base it on the `glostylesuper` style:

```
8005 \setglossarystyle{super}%
```

Put the glossary in a `supertabular` environment with two columns, a header and horizontal lines above and below the table:

```
8006 \renewenvironment{theglossary}%
8007   {\tablehead{\hline\bfseries \entryname &
8008     \bfseries \descriptionname\tabularnewline\hline}%
8009   \tabletail{\hline}
8010   \begin{supertabular}{|lp{\glsdescwidth}|}%
8011   {\end{supertabular}}%
8012 }
```

`super3col` The `super3col` style is like the `super` style, but with 3 columns:

```
8013 \newglossarystyle{super3col}{%
```

Put the glossary in a `supertabular` environment with three columns and no head or tail:

```
8014 \renewenvironment{theglossary}%
8015   {\tablehead{}\tabletail{}}%
8016   \begin{supertabular}{lp{\glsdescwidth}p{\glspagelistwidth}}%
8017   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8018 \renewcommand*\glossaryheader{}
```

No group headings:

```
8019 \renewcommand*\glsgroupheading[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
8020 \renewcommand{\glossentry}[2]{%
8021   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8022   \glossentrydesc{##1} & ##2\tabularnewline
8023 }
```

Sub entries on a row (no name, description in second column, page list in last column):

```
8024 \renewcommand{\subglossentry}[3]{%
8025     &
8026     \glssubentryitem{##2}%
8027     \glstarget{##2}{\strut}\glossentrydesc{##2} &
8028     ##3\tabularnewline
8029 }%
```

Blank row between groups:

```
8030 \renewcommand*{\glsgroupskip}{%
8031     \ifglsnogroupskip\else & &\tabularnewline\fi}%
8032 }
```

`super3colborder` The `super3colborder` style is like the `super3col` style, but with a border:

```
8033 \newglossarystyle{super3colborder}{%
    Base it on the glostylesuper3col style:
8034 \setglossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns and a horizontal line in the head and tail:
8035 \renewenvironment{theglossary}%
8036     {\tablehead{\hline}\tabletail{\hline}%
8037     \begin{supertabular}{|l|p{\glsdescwidth}|p{\glspagelistwidth}|}%
8038     {\end{supertabular}}%
8039 }
```

`super3colheader` The `super3colheader` style is like the `super3col` style but with a header row:

```
8040 \newglossarystyle{super3colheader}{%
    Base it on the glostylesuper3col style:
8041 \setglossarystyle{super3col}%
    Put the glossary in a supertabular environment with three columns, a header and no tail:
8042 \renewenvironment{theglossary}%
8043     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8044         \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8045     \begin{supertabular}{lp{\glsdescwidth}p{\glspagelistwidth}}%
8046     {\end{supertabular}}%
8047 }
```

`super3colheaderborder` The `super3colheaderborder` style is like the `super3col` style but with a header and border:

```
8048 \newglossarystyle{super3colheaderborder}{%
    Base it on the glostylesuper3colborder style:
8049 \setglossarystyle{super3colborder}%
```


Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```

8050 \renewenvironment{theglossary}%
8051   {\tablehead{\hline
8052     \bfseries\entryname&\bfseries\descriptionname&
8053     \bfseries\pagelistname\tabularnewline\hline}%
8054   \tabletail{\hline}%
8055   \begin{supertabular}{|l|p{\glsdescwidth}|p{\glspagelistwidth}|}%
8056   {\end{supertabular}}%
8057 }

```

`super4col` The `super4col` glossary style has four columns, where the third column contains the value of the corresponding symbol key used when that entry was defined.

```

8058 \newglossarystyle{super4col}{%

```

Put the glossary in a supertabular environment with four columns and no head or tail:

```

8059 \renewenvironment{theglossary}%
8060   {\tablehead{}\tabletail{}%
8061   \begin{supertabular}{|l|l|l|l|}%
8062   \end{supertabular}}%

```

Do nothing at the start of the table:

```

8063 \renewcommand*{\glossaryheader}{}%

```

No group headings:

```

8064 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entries on a row with the name in the first column, description in second column, symbol in third column and page list in last column:

```

8065 \renewcommand{\glossentry}[2]{%
8066   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8067   \glossentrydesc{##1} &
8068   \glossentrysymbol{##1} & ##3\tabularnewline
8069 }%

```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```

8070 \renewcommand{\subglossentry}[3]{%
8071   &
8072   \glsesubentryitem{##2}%
8073   \glstarget{##2}{\strut}\glossentrydesc{##2} &
8074   \glossentrysymbol{##2} & ##3\tabularnewline
8075 }%

```

Blank row between groups:

```

8076 \renewcommand*{\glsgroupskip}{%
8077   \ifglsnogroupskip\else & & \tabularnewline\fi}%
8078 }

```

`super4colheader` The `super4colheader` style is like the `super4col` but with a header row.

```
8079 \newglossarystyle{super4colheader}{%
      Base it on the glostylesuper4col style:
8080 \setglossarystyle{super4col}%
      Put the glossary in a supertabular environment with four columns, a header and
      no tail:
8081 \renewenvironment{theglossary}%
8082   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8083     \bfseries\symbolname &
8084     \bfseries\pagelistname\tabularnewline}%
8085   \tabletail{}}%
8086   \begin{supertabular}{|l|l|l|l|}%
8087   {\end{supertabular}}%
8088 }
```

`super4colborder` The `super4colborder` style is like the `super4col` but with a border.

```
8089 \newglossarystyle{super4colborder}{%
      Base it on the glostylesuper4col style:
8090 \setglossarystyle{super4col}%
      Put the glossary in a supertabular environment with four columns and a hori-
      zontal line in the head and tail:
8091 \renewenvironment{theglossary}%
8092   {\tablehead{\hline}\tabletail{\hline}%
8093   \begin{supertabular}{|l|l|l|l|}%
8094   {\end{supertabular}}%
8095 }
```

`super4colheaderborder` The `super4colheaderborder` style is like the `super4col` but with a header and border.

```
8096 \newglossarystyle{super4colheaderborder}{%
      Base it on the glostylesuper4col style:
8097 \setglossarystyle{super4col}%
      Put the glossary in a supertabular environment with four columns and a header
      bordered by horizontal lines and a horizontal line in the tail:
8098 \renewenvironment{theglossary}%
8099   {\tablehead{\hline\bfseries\entryname&\bfseries\descriptionname&
8100     \bfseries\symbolname &
8101     \bfseries\pagelistname\tabularnewline\hline}%
8102   \tabletail{\hline}%
8103   \begin{supertabular}{|l|l|l|l|}%
8104   {\end{supertabular}}%
8105 }
```

`altsuper4col` The `altsuper4col` glossary style is like `super4col` but has provision for multiline descriptions.

```
8106 \newglossarystyle{altsuper4col}{%
      Base it on the glostylesuper4col style:
8107 \setglossarystyle{super4col}%
      Put the glossary in a supertabular environment with four columns and no head
      or tail:
8108 \renewenvironment{theglossary}%
8109   {\tablehead{}\tabletail{}}%
8110   \begin{supertabular}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
8111   {\end{supertabular}}}%
8112 }
```

`altsuper4colheader` The `altsuper4colheader` style is like the `altsuper4col` but with a header row.

```
8113 \newglossarystyle{altsuper4colheader}{%
      Base it on the glostylesuper4colheader style:
8114 \setglossarystyle{super4colheader}%
      Put the glossary in a supertabular environment with four columns, a header and
      no tail:
8115 \renewenvironment{theglossary}%
8116   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8117     \bfseries\symbolname &
8118     \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8119   \begin{supertabular}{lp{\glsdescwidth}lp{\glspagelistwidth}}}%
8120   {\end{supertabular}}}%
8121 }
```

`altsuper4colborder` The `altsuper4colborder` style is like the `altsuper4col` but with a border.

```
8122 \newglossarystyle{altsuper4colborder}{%
      Base it on the glostylesuper4colborder style:
8123 \setglossarystyle{super4colborder}%
      Put the glossary in a supertabular environment with four columns and a hori-
      zontal line in the head and tail:
8124 \renewenvironment{theglossary}%
8125   {\tablehead{\hline}\tabletail{\hline}%
8126   \begin{supertabular}%
8127     {||lp{\glsdescwidth}lp{\glspagelistwidth}||}}%
8128   {\end{supertabular}}}%
8129 }
```

`altsuper4colheaderborder` The `altsuper4colheaderborder` style is like the `altsuper4col` but with a header and border.

```
8130 \newglossarystyle{altsuper4colheaderborder}{%
```

Base it on the `glostylesuper4colheaderborder` style:

```
8131 \setglossarystyle{super4colheaderborder}%
```

Put the glossary in a `supertabular` environment with four columns and a header bordered by horizontal lines and a horizontal line in the tail:

```
8132 \renewenvironment{theglossary}%
8133   {\tablehead{\hline
8134     \bfseries\entryname &
8135     \bfseries\descriptionname &
8136     \bfseries\symbolname &
8137     \bfseries\pagelistname\tabularnewline\hline}%
8138   \tabletail{\hline}%
8139   \begin{supertabular}%
8140     {\lllp{\glsdescwidth}|llp{\glspagelistwidth}|}%
8141   {\end{supertabular}}%
8142 }
```

5.8 Glossary Styles using `supertabular` environment (`glossary-superragged` package)

The glossary styles defined in the package use the `supertabular` environment. These styles are like those provided by the package, except that the multiline columns have ragged right justification.

```
8143 \ProvidesPackage{glossary-superragged}[2013/11/14 v4.0 (NLCT)]
```

Requires the package:

```
8144 \RequirePackage{array}
```

Requires the package:

```
8145 \RequirePackage{supertabular}
```

`\glsdescwidth` This is a length that governs the width of the description column. This may already have been defined.

```
8146 \@ifundefined{glsdescwidth}{%
8147   \newlength\glsdescwidth
8148   \setlength{\glsdescwidth}{0.6\hsize}
8149 }{}
```

`\glspagelistwidth` This is a length that governs the width of the page list column. This may already have been defined.

```
8150 \@ifundefined{glspagelistwidth}{%
8151   \newlength\glspagelistwidth
8152   \setlength{\glspagelistwidth}{0.1\hsize}
8153 }{}
```

`superragged` The `superragged` glossary style uses the `supertabular` environment.

```
8154 \newglossarystyle{superragged}{%
```

Put the glossary in a supertabular environment with two columns and no head or tail:

```
8155 \renewenvironment{theglossary}%
8156   {\tablehead{}\tabletail{}}%
8157   \begin{supertabular}{|l>{\raggedright}p{\glsdescwidth}}%
8158   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8159 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8160 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries put in a row (name in first column, description and page list in second column):

```
8161 \renewcommand{\glossentry}[2]{%
8162   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8163   \glossentrydesc{##1}\glspostdescription\space ##2%
8164   \tabularnewline
8165 }%
```

Sub entries put in a row (no name, description and page list in second column):

```
8166 \renewcommand{\subglossentry}[3]{%
8167   &
8168   \glssubentryitem{##2}%
8169   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription\space
8170   ##3%
8171   \tabularnewline
8172 }%
```

Blank row between groups:

```
8173 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
8174 }
```

superraggedborder The superraggedborder style is like the above, but with horizontal and vertical lines:

```
8175 \newglossarystyle{superraggedborder}{%
```

Base it on the glostylesuperragged style:

```
8176 \setglossarystyle{superragged}%
```

Put the glossary in a supertabular environment with two columns and a horizontal line in the head and tail:

```
8177 \renewenvironment{theglossary}%
8178   {\tablehead{\hline}\tabletail{\hline}}%
8179   \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|}%
8180   {\end{supertabular}}%
8181 }
```

superraggedheader The superraggedheader style is like the super style, but with a header:

```
8182 \newglossarystyle{superraggedheader}{%
```

Base it on the `glostylesuperragged` style:

```
8183 \setglossarystyle{superragged}%
```

Put the glossary in a `supertabular` environment with two columns, a header and no tail:

```
8184 \renewenvironment{theglossary}%
8185   {\tablehead{\bfseries \entryname & \bfseries \descriptionname
8186     \tabularnewline}%
8187   \tabletail{}}%
8188   \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}}%
8189   {\end{supertabular}}%
8190 }
```

`superraggedheaderborder` The `superraggedheaderborder` style is like the `superragged` style but with a header and border:

```
8191 \newglossarystyle{superraggedheaderborder}{%
```

Base it on the `glostylesuper` style:

```
8192 \setglossarystyle{superragged}%
```

Put the glossary in a `supertabular` environment with two columns, a header and horizontal lines above and below the table:

```
8193 \renewenvironment{theglossary}%
8194   {\tablehead{\hline\bfseries \entryname &
8195     \bfseries \descriptionname\tabularnewline\hline}%
8196   \tabletail{\hline}
8197   \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}||}%
8198   {\end{supertabular}}%
8199 }
```

`superragged3col` The `superragged3col` style is like the `superragged` style, but with 3 columns:

```
8200 \newglossarystyle{superragged3col}{%
```

Put the glossary in a `supertabular` environment with three columns and no head or tail:

```
8201 \renewenvironment{theglossary}%
8202   {\tablehead{}\tabletail{}}%
8203   \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}%
8204     >{\raggedright}p{\glspagerlistwidth}}%
8205   {\end{supertabular}}%
```

Do nothing at the start of the table:

```
8206 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8207 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row (name in first column, description in second column, page list in last column):

```
8208 \renewcommand{\glossentry}[2]{}%
8209 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
```

```

8210 \glossentrydesc{##1} &
8211 ##2\tabularnewline
8212 }%

```

Sub entries on a row (no name, description in second column, page list in last column):

```

8213 \renewcommand{\subglossentry}[3]{%
8214     &
8215     \glssubentryitem{##2}%
8216     \glstarget{##2}{\strut}\glossentrydesc{##2} &
8217     ##3\tabularnewline
8218 }%

```

Blank row between groups:

```

8219 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & \tabularnewline\fi}%
8220 }

```

`superragged3colborder` The `superragged3colborder` style is like the `superragged3col` style, but with a border:

```

8221 \newglossarystyle{superragged3colborder}{%

```

Base it on the `glostylesuperragged3col` style:

```

8222 \setglossarystyle{superragged3col}%

```

Put the glossary in a `supertabular` environment with three columns and a horizontal line in the head and tail:

```

8223 \renewenvironment{theglossary}%
8224     {\tablehead{\hline}\tabletail{\hline}}%
8225     \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|%
8226         >{\raggedright}p{\glspagelistwidth}|}%
8227     {\end{supertabular}}%
8228 }

```

`superragged3colheader` The `superragged3colheader` style is like the `superragged3col` style but with a header row:

```

8229 \newglossarystyle{superragged3colheader}{%

```

Base it on the `glostylesuperragged3col` style:

```

8230 \setglossarystyle{superragged3col}%

```

Put the glossary in a `supertabular` environment with three columns, a header and no tail:

```

8231 \renewenvironment{theglossary}%
8232     {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8233         \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8234     \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}%
8235         >{\raggedright}p{\glspagelistwidth}}%
8236     {\end{supertabular}}%
8237 }

```

ght3colheaderborder The superragged3colheaderborder style is like the superragged3col style but with a header and border:

```
8238 \newglossarystyle{superragged3colheaderborder}{%
```

Base it on the glostylesuperragged3colborder style:

```
8239 \setglossarystyle{superragged3colborder}{%
```

Put the glossary in a supertabular environment with three columns, a header with horizontal lines and a horizontal line in the tail:

```
8240 \renewenvironment{theglossary}{%
8241   {\tablehead{\hline
8242     \bfseries\entryname&\bfseries\descriptionname&
8243     \bfseries\pagelistname\tabularnewline\hline}%
8244   \tabletail{\hline}%
8245   \begin{supertabular}{|l|>{\raggedright}p{\glsdescwidth}|%
8246     >{\raggedright}p{\glspagelistwidth}|}%
8247   {\end{supertabular}}}%
8248 }
```

altsuperragged4col The altsuperragged4col glossary style is like altsuper4col style in the package but uses ragged right formatting in the description and page list columns.

```
8249 \newglossarystyle{altsuperragged4col}{%
```

Put the glossary in a supertabular environment with four columns and no head or tail:

```
8250 \renewenvironment{theglossary}{%
8251   {\tablehead{}\tabletail{}%
8252   \begin{supertabular}{|l>{\raggedright}p{\glsdescwidth}l%
8253     >{\raggedright}p{\glspagelistwidth}}}%
8254   {\end{supertabular}}}%
```

Do nothing at the start of the table:

```
8255 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8256 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries on a row with the name in the first column, description in second column, symbol in third column and page list in last column:

```
8257 \renewcommand{\glossentry}[2]{%
8258   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
8259   \glossentrydesc{##1} &
8260   \glossentrysymbol{##1} & ##2\tabularnewline
8261 }%
```

Sub entries on a row with no name, the description in the second column, symbol in third column and page list in last column:

```
8262 \renewcommand{\subglossentry}[3]{%
8263   &
8264   \glssubentryitem{##2}%
8265   \glstarget{##2}{\strut}\glossentrydesc{##2} &
```



```
8266     \glossentrysymbol{##2} & ##3\tabularnewline
8267 }%
```

Blank row between groups:

```
8268 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else & &\tabularnewline\fi}%
8269 }
```

`altperragged4colheader` The `altperragged4colheader` style is like the `altperragged4col` style but with a header row.

```
8270 \newglossarystyle{altperragged4colheader}{%
```

Base it on the `glostylealtperragged4col` style:

```
8271 \setglossarystyle{altperragged4col}%
```

Put the glossary in a `supertabular` environment with four columns, a header and no tail:

```
8272 \renewenvironment{theglossary}%
8273   {\tablehead{\bfseries\entryname&\bfseries\descriptionname&
8274             \bfseries\symbolname &
8275             \bfseries\pagelistname\tabularnewline}\tabletail{}}%
8276   \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}1%
8277     >{\raggedright}p{\glspagelistwidth}}}%
8278   {\end{supertabular}}%
8279 }
```

`altperragged4colborder` The `altperragged4colborder` style is like the `altperragged4col` style but with a border.

```
8280 \newglossarystyle{altperragged4colborder}{%
```

Base it on the `glostylealtperragged4col` style:

```
8281 \setglossarystyle{altperragged4col}%
```

Put the glossary in a `supertabular` environment with four columns and a horizontal line in the head and tail:

```
8282 \renewenvironment{theglossary}%
8283   {\tablehead{\hline}\tabletail{\hline}%
8284   \begin{supertabular}%
8285     {|1|>{\raggedright}p{\glsdescwidth}|1|%
8286     >{\raggedright}p{\glspagelistwidth}|}}%
8287   {\end{supertabular}}%
8288 }
```

`altperragged4colheaderborder` The `altperragged4colheaderborder` style is like the `altperragged4col` style but with a header and border.

```
8289 \newglossarystyle{altperragged4colheaderborder}{%
```

Base it on the `glostylealtperragged4col` style:

```
8290 \setglossarystyle{altperragged4col}%
```

Put the glossary in a supertabular environment with four columns and a header bordered by horizontal lines and a horizontal line in the tail:

```

8291 \renewenvironment{theglossary}%
8292   {\tablehead{\hline
8293     \bfseries\entryname &
8294     \bfseries\descriptionname &
8295     \bfseries\symbolname &
8296     \bfseries\pagelistname\tabularnewline\hline}%
8297   \tabletail{\hline}%
8298   \begin{supertabular}%
8299     {|||>{\raggedright}p{\glsdescwidth}|1|}
8300     >{\raggedright}p{\glspagelistwidth}|}}%
8301   {\end{supertabular}}%
8302 }

```

5.9 Tree Styles (glossary-tree.sty)

The style file defines glossary styles that have a tree-like structure. These are designed for hierarchical glossaries.

```

8303 \ProvidesPackage{glossary-tree}[2014/08/27 v4.10 (NLCT)]

```

`\glstreenamefmt` Format used to display the name in the tree styles. (This may be counteracted by `\glsnamefont`.) This command is also used to format the group headings.

```

8304 \newcommand*{\glstreenamefmt}[1]{\textbf{#1}}

```

`index` The index glossary style is similar in style to the way indices are usually typeset using `\item`, `\subitem` and `\subsubitem`. The entry name is set in bold. If an entry has a symbol, it is placed in brackets after the name. Then the description is displayed, followed by the number list. This style allows up to three levels.

```

8305 \newglossarystyle{index}{%

```

Set the paragraph indentation and skip and define `\item` to be the same as that used by the index:

```

8306 \renewenvironment{theglossary}%
8307   {\setlength{\parindent}{0pt}%
8308   \setlength{\parskip}{0pt plus 0.3pt}%
8309   \let\item\@idxitem}%
8310   {\par}%

```

Do nothing at the start of the environment:

```

8311 \renewcommand*{\glossaryheader}{}%

```

No group headers:

```

8312 \renewcommand*{\glsgroupheading}[1]{}%

```

Main (level 0) entry starts a new item with the name in bold followed by the symbol in brackets (if it exists), the description and the page list.

```

8313 \renewcommand*{\glossentry}[2]{%

```

```

8314     \item\glstentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
8315     \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8316     \space \glossentrydesc{##1}\glspostdescription\space ##2%
8317 }%

```

Sub entries: level 1 entries use `\subitem`, levels greater than 1 use `\subsubitem`. The level (`##1`) shouldn't be 0, as that's catered by `\glossentry`, but for completeness, if the level is 0, `\item` is used. The name is put in bold, followed by the symbol in brackets (if it exists), the description and the page list.

```

8318 \renewcommand{\subglossentry}[3]{%
8319   \ifcase##1\relax
8320   % level 0
8321   \item
8322   \or
8323   % level 1
8324   \subitem
8325   \glssubentryitem{##2}%
8326   \else
8327   % all other levels
8328   \subsubitem
8329   \fi
8330   \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
8331   \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
8332   \space\glossentrydesc{##2}\glspostdescription\space ##3%
8333 }%

```

Vertical gap between groups is the same as that used by indices:

```

8334 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}

```

`indexgroup` The `indexgroup` style is like the `index` style but has headings.

```

8335 \newglossarystyle{indexgroup}{%

```

Base it on the `glostyleindex` style:

```

8336 \setglossarystyle{index}%

```

Add a heading for each group. This puts the group's title in bold followed by a vertical gap.

```

8337 \renewcommand*{\glsgroupheading}[1]{%
8338   \item\glstreenamefmt{\glsgrouptitle{##1}}\indexspace}%
8339 }

```

`indexhypergroup` The `indexhypergroup` style is like the `indexgroup` style but has hyper navigation.

```

8340 \newglossarystyle{indexhypergroup}{%

```

Base it on the `glostyleindex` style:

```

8341 \setglossarystyle{index}%

```

Put navigation links to the groups at the start of the glossary:

```

8342 \renewcommand*{\glossaryheader}{%
8343   \item\glstreenamefmt{\glsnavigation}\indexspace}%

```

Add a heading for each group (with a target). The group's title is in bold followed by a vertical gap.

```
8344 \renewcommand*{\glsgroupheading}[1]{%
8345   \item\glstreenamefmt{\glsnavhypertarget{##1}{\glssetgrouptitle{##1}}}%
8346   \indexspace}%
8347 }
```

tree The tree glossary style is similar in style to the index style, but can have arbitrary levels.

```
8348 \newglossarystyle{tree}{%
```

Set the paragraph indentation and skip:

```
8349 \renewenvironment{theglossary}%
8350   {\setlength{\parindent}{0pt}%
8351   \setlength{\parskip}{0pt plus 0.3pt}}%
8352   {}%
```

Do nothing at the start of the theglossary environment:

```
8353 \renewcommand*{\glossaryheader}{}%
```

No group headings:

```
8354 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries: name in bold, followed by symbol in brackets (if it exists), the description and the page list:

```
8355 \renewcommand{\glossentry}[2]{%
8356   \hangindent0pt\relax
8357   \parindent0pt\relax
8358   \glsentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
8359   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8360   \space\glossentrydesc{##1}\glspostdescription\space##2\par
8361   }%
```

Sub entries: level $\langle n \rangle$ is indented by $\langle n \rangle$ times `\glstreeindent`. The name is in bold, followed by the symbol in brackets (if it exists), the description and the page list.

```
8362 \renewcommand{\subglossentry}[3]{%
8363   \hangindent##1\glstreeindent\relax
8364   \parindent##1\glstreeindent\relax
8365   \ifnum##1=1\relax
8366     \glssubentryitem{##2}%
8367     \fi
8368     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
8369     \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
8370     \space\glossentrydesc{##2}\glspostdescription\space ##3\par
8371   }%
```

Vertical gap between groups is the same as that used by indices:

```
8372 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}}
```

`treegroup` Like the tree style but the glossary groups have headings.

```
8373 \newglossarystyle{treegroup}{%
```

```
Base it on the glostyletree style:
```

```
8374 \setglossarystyle{tree}{%
```

```
Each group has a heading (in bold) followed by a vertical gap):
```

```
8375 \renewcommand{\glsgroupheading}[1]{\par
```

```
8376 \noindent\glstreenamefmt{\glsgetgrouptitle{##1}}\par\indexspace}%
```

```
8377 }
```

`treehypergroup` The `treehypergroup` style is like the `treegroup` style, but has a set of links to the groups at the start of the glossary.

```
8378 \newglossarystyle{treehypergroup}{%
```

```
Base it on the glostyletree style:
```

```
8379 \setglossarystyle{tree}{%
```

```
Put navigation links to the groups at the start of the theglossary environment:
```

```
8380 \renewcommand*{\glossaryheader}{%
```

```
8381 \par\noindent\glstreenamefmt{\glsnavigation}\par\indexspace}%
```

```
Each group has a heading (in bold with a target) followed by a vertical gap):
```

```
8382 \renewcommand*{\glsgroupheading}[1]{%
```

```
8383 \par\noindent
```

```
8384 \glstreenamefmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
```

```
8385 \indexspace}%
```

```
8386 }
```

`\glstreeindent` Length governing left indent for each level of the tree style.

```
8387 \newlength\glstreeindent
```

```
8388 \setlength{\glstreeindent}{10pt}
```

`treenoname` The `treenoname` glossary style is like the tree style, but doesn't print the name or symbol for sub-levels.

```
8389 \newglossarystyle{treenoname}{%
```

```
Set the paragraph indentation and skip:
```

```
8390 \renewenvironment{theglossary}{%
```

```
8391 {\setlength{\parindent}{0pt}%
```

```
8392 \setlength{\parskip}{0pt plus 0.3pt}}%
```

```
8393 {}%
```

```
No header:
```

```
8394 \renewcommand*{\glossaryheader}{}%
```

```
No group headings:
```

```
8395 \renewcommand*{\glsgroupheading}[1]{}%
```

Main (level 0) entries: the name is in bold, followed by the symbol in brackets (if it exists), the description and the page list.

```

8396 \renewcommand{\glossentry}[2]{%
8397   \hangindent0pt\relax
8398   \parindent0pt\relax
8399   \glstentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
8400   \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
8401   \space\glossentrydesc{##1}\glspostdescription\space##2\par
8402 }%

```

Sub entries: level $\langle n \rangle$ is indented by $\langle n \rangle$ times `\glstreeindent`. The name and symbol are omitted. The description followed by the page list are displayed.

```

8403 \renewcommand{\subglossentry}[3]{%
8404   \hangindent##1\glstreeindent\relax
8405   \parindent##1\glstreeindent\relax
8406   \ifnum##1=1\relax
8407     \glssubentryitem{##2}%
8408     \fi
8409     \glstarget{##2}{\strut}%
8410     \glossentrydesc{##2}\glspostdescription\space##3\par
8411 }%

```

Vertical gap between groups is the same as that used by indices:

```

8412 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
8413 }

```

`treenonamegroup` Like the `treenoname` style but the glossary groups have headings.

```
8414 \newglossarystyle{treenonamegroup}{%
```

Base it on the `glostyletreenoname` style:

```
8415 \setglossarystyle{treenoname}{%
```

Give each group a heading:

```

8416 \renewcommand{\glsgroupheading}[1]{\par
8417   \noindent\glstreenamefmt{\glsgrouptitle{##1}}\par\indexspace}%
8418 }

```

`treenonamehypergroup` The `treenonamehypergroup` style is like the `treenonamegroup` style, but has a set of links to the groups at the start of the glossary.

```
8419 \newglossarystyle{treenonamehypergroup}{%
```

Base it on the `glostyletreenoname` style:

```
8420 \setglossarystyle{treenoname}{%
```

Put navigation links to the groups at the start of the `theglossary` environment:

```

8421 \renewcommand*{\glossaryheader}{%
8422   \par\noindent\glstreenamefmt{\glsnavigation}\par\indexspace}%

```

Each group has a heading (in bold with a target) followed by a vertical gap):

```

8423 \renewcommand*{\glsgroupheading}[1]{%
8424   \par\noindent

```

```

8425 \glstreenamefmt{\glstreehypertarget{##1}{\glstreegroupname{##1}}}\par
8426 \indexspace}%
8427 }

```

`\glsssetwidest` `\glsssetwidest[level]{text}` sets the widest text for the given level. It is used by the `almtree` glossary styles to determine the indentation of each level.

```

8428 \newcommand*{\glsssetwidest}[2][0]{%
8429 \expandafter\def\csname @glswidestname\romannumeral#1\endcsname{%
8430 #2}%
8431 }

```

`\@glswidestname` Initialise `\@glswidestname`.

```

8432 \newcommand*{\@glswidestname}{}

```

`almtree` The `almtree` glossary style is similar in style to the `tree` style, but the indentation is obtained from the width of `\@glswidestname` which is set using `\glsssetwidest`.

```

8433 \newglossarystyle{almtree}{%

```

Redefine the `theglossary` environment.

```

8434 \renewenvironment{theglossary}%
8435 {\def\@gls@prevlevel{-1}%
8436 \mbox{}\par}%
8437 {\par}%

```

Set the header and group headers to nothing.

```

8438 \renewcommand*{\glossaryheader}{}%
8439 \renewcommand*{\glsgroupheading}[1]{}%

```

Redefine the way that the level 0 entries are displayed.

```

8440 \renewcommand{\glossentry}[2]{%
8441 \ifnum\@gls@prevlevel=0\relax
8442 \else

```

Find out how big the indentation should be by measuring the widest entry.

```

8443 \settowidth{\glstreeindent}{\glstreenamefmt{\@glswidestname\space}}%
8444 \fi

```

Set the `hangindent` and `paragraph indent`.

```

8445 \hangindent\glstreeindent
8446 \parindent\glstreeindent

```

Put the name to the left of the paragraph block.

```

8447 \makebox[0pt][r]{\makebox[\glstreeindent][l]}%
8448 \glsentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%

```

If the symbol is missing, ignore it, otherwise put it in brackets.

```

8449 \ifglshassymbol{##1}{(\glossentrysymbol{##1})\space}{}%

```

Do the description followed by the description terminator and location list.

```

8450 \glossentrydesc{##1}\glspostdescription \space ##2\par

```

Set the previous level to 0.

```
8451 \def\@gls@prevlevel{0}%  
8452 }%
```

Redefine the way sub-entries are displayed.

```
8453 \renewcommand{\subglossentry}[3]{%
```

Increment and display the sub-entry counter if this is a level 1 entry and the sub-entry counter is in use.

```
8454 \ifnum##1=1\relax  
8455 \glsentryitem{##2}%  
8456 \fi
```

If the level hasn't changed, keep the same settings, otherwise adjust `\glstreeindent` accordingly.

```
8457 \ifnum\@gls@prevlevel=##1\relax  
8458 \else
```

Compute the widest entry for this level, or for level 0 if not defined for this level.

Store in `\gls@tmplen`

```
8459 \@ifundefined{glswidestname\romannumeral##1}{%  
8460 \settowidth{\gls@tmplen}{\glstreenamfmt{\@glswidestname\space}}}%  
8461 \settowidth{\gls@tmplen}{\glstreenamfmt{%  
8462 \csname @glswidestname\romannumeral##1\endcsname\space}}}%
```

Determine if going up or down a level

```
8463 \ifnum\@gls@prevlevel<##1\relax
```

Depth has increased, so add the width of the widest entry to `\glstreeindent`.

```
8464 \setlength\glstreeindent\gls@tmplen  
8465 \addtolength\glstreeindent\parindent  
8466 \parindent\glstreeindent  
8467 \else
```

Depth has decreased, so subtract width of the widest entry from the previous level to `\glstreeindent`. First determine the width of the widest entry for the previous level and store in `\glstreeindent`.

```
8468 \@ifundefined{glswidestname\romannumeral\@gls@prevlevel}{%  
8469 \settowidth{\glstreeindent}{\glstreenamfmt{%  
8470 \@glswidestname\space}}}%  
8471 \settowidth{\glstreeindent}{\glstreenamfmt{%  
8472 \csname @glswidestname\romannumeral\@gls@prevlevel  
8473 \endcsname\space}}}%
```

Subtract this length from the previous level's paragraph indent and set to `\glstreeindent`.

```
8474 \addtolength\parindent{-\glstreeindent}%  
8475 \setlength\glstreeindent\parindent  
8476 \fi  
8477 \fi
```


Set the hanging indentation.

```
8478 \hangindent\glstreeindent
```

Put the name to the left of the paragraph block

```
8479 \makebox[Opt][r]{\makebox[\gls@tmplen][1]{%  
8480 \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
```

If the symbol is missing, ignore it, otherwise put it in brackets.

```
8481 \ifglshassymbol{##2}{(\glossentrysymbol{##2})\space}{}%
```

Do the description followed by the description terminator and location list.

```
8482 \glossentrydesc{##2}\glspostdescription\space ##3\par
```

Set the previous level macro to the current level.

```
8483 \def\@gls@prevlevel{##1}%  
8484 }%
```

Vertical gap between groups is the same as that used by indices:

```
8485 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%  
8486 }
```

almtreegroup Like the almtree style but the glossary groups have headings.

```
8487 \newglossarystyle{almtreegroup}{%
```

Base it on the glostylealmtree style:

```
8488 \setglossarystyle{almtree}%
```

Give each group a heading.

```
8489 \renewcommand{\glsgroupheading}[1]{\par  
8490 \def\@gls@prevlevel{-1}%  
8491 \hangindentOpt\relax  
8492 \parindentOpt\relax  
8493 \glstreenamefmt{\glsgetgrouptitle{##1}}\par\indexspace}%  
8494 }
```

almtreehypergroup The almtreehypergroup style is like the almtreegroup style, but has a set of links to the groups at the start of the glossary.

```
8495 \newglossarystyle{almtreehypergroup}{%
```

Base it on the glostylealmtree style:

```
8496 \setglossarystyle{almtree}%
```

Put the navigation links in the header

```
8497 \renewcommand*{\glossaryheader}{%  
8498 \par  
8499 \def\@gls@prevlevel{-1}%  
8500 \hangindentOpt\relax  
8501 \parindentOpt\relax  
8502 \glstreenamefmt{\glsnavigation}\par\indexspace}%
```

Put a hypertarget at the start of each group

```
8503 \renewcommand*\glsgroupheading}[1]{%
8504   \par
8505   \def\@gls@prevlevel{-1}%
8506   \hangindent0pt\relax
8507   \parindent0pt\relax
8508   \glstreenamfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}\par
8509   \indexspace}}
```

6 glossaries-compatible-207

Provides compatibility with version 2.07 and below. This uses original glossaries xindy and makeindex formatting, so can be used with old documents that had customized style files, but hyperlinks may not work properly.

```
8510 \NeedsTeXFormat{LaTeX2e}
8511 \ProvidesPackage{glossaries-compatible-207}[2011/04/02 v1.0 (NLCT)]
```

`\GlsAddXdyAttribute` Adds an attribute in old format.

```
8512 \ifglsxindy
8513   \renewcommand*\GlsAddXdyAttribute[1]{%
8514     \edef\@xdyattributes{\@xdyattributes ^^J \string"#1\string"%}
8515     \expandafter\toks@\expandafter{\@xdylocref}%
8516     \edef\@xdylocref{\the\toks@ ^^J%
8517       (markup-locref
8518       :open \string"\string~\string\setentrycounter
8519         {\noexpand\glscounter}%
8520         \expandafter\string\csname#1\endcsname
8521         \expandafter\@gobble\string\{\string" ^^J
8522         :close \string"\expandafter\@gobble\string\}\string" ^^J
8523         :attr \string"#1\string")}}
```

Only has an effect before `\writeist`:

```
8524 \fi
```

`\GlsAddXdyCounters`

```
8525 \renewcommand*\GlsAddXdyCounters[1]{%
8526   \GlossariesWarning{\string\GlsAddXdyCounters\space not available
8527     in compatibility mode.}%
8528 }
```

Add predefined attributes

```
8529 \GlsAddXdyAttribute{glsnumberformat}
8530 \GlsAddXdyAttribute{textrm}
8531 \GlsAddXdyAttribute{textsf}
8532 \GlsAddXdyAttribute{texttt}
8533 \GlsAddXdyAttribute{textbf}
8534 \GlsAddXdyAttribute{textmd}
8535 \GlsAddXdyAttribute{textit}
```

```

8536 \GlsAddXdyAttribute{textup}
8537 \GlsAddXdyAttribute{textsl}
8538 \GlsAddXdyAttribute{textsc}
8539 \GlsAddXdyAttribute{emph}
8540 \GlsAddXdyAttribute{glshypernumber}
8541 \GlsAddXdyAttribute{hyperrrm}
8542 \GlsAddXdyAttribute{hypersf}
8543 \GlsAddXdyAttribute{hypertt}
8544 \GlsAddXdyAttribute{hyperbf}
8545 \GlsAddXdyAttribute{hypermd}
8546 \GlsAddXdyAttribute{hyperit}
8547 \GlsAddXdyAttribute{hyperup}
8548 \GlsAddXdyAttribute{hypersl}
8549 \GlsAddXdyAttribute{hypersc}
8550 \GlsAddXdyAttribute{hyperemph}

```

\GlsAddXdyLocation Restore v2.07 definition:

```

8551 \ifglxindy
8552   \renewcommand*{\GlsAddXdyLocation}[2]{%
8553     \edef\xdyuserlocationdefs{%
8554       \xdyuserlocationdefs ^^J%
8555       (define-location-class \string"#1\string"^^J\space\space
8556       \space(#2))
8557     }%
8558     \edef\xdyuserlocationnames{%
8559       \xdyuserlocationnames^^J\space\space\space
8560       \string"#1\string"}%
8561   }
8562 \fi

```

\@do@wrglossary

```

8563 \renewcommand{\@do@wrglossary}[1]{%
  Determine whether to use xindy or makeindex syntax
8564 \ifglxindy
  Need to determine if the formatting information starts with a ( or ) indicating a
  range.
8565   \expandafter\@glo@check@mkidxrangechar\@glsnumberformat\@nil
8566   \def\@glo@range{}%
8567   \expandafter\if\@glo@prefix(\relax
8568     \def\@glo@range{:open-range}%
8569   \else
8570     \expandafter\if\@glo@prefix)\relax
8571     \def\@glo@range{:close-range}%
8572   \fi
8573 \fi

  Get the location and escape any special characters
8574   \protected@edef\@glslocref{\theglstrycounter}%
8575   \@gls@checkmkidxchars\@glslocref

```

Write to the glossary file using xindy syntax.

```
8576 \glossary[\csname glo@#1@type\endcsname]{%
8577 (indexentry :tkey (\csname glo@#1@index\endcsname)
8578 :locoref \string"\@glslocoref\string" %
8579 :attr \string"\@glo@suffix\string" \@glo@range
8580 )
8581 }%
8582 \else
```

Convert the format information into the format required for makeindex

```
8583 \@set@glo@numformat\@glo@numfmt\@gls@counter\@glsnumberformat
```

Write to the glossary file using makeindex syntax.

```
8584 \glossary[\csname glo@#1@type\endcsname]{%
8585 \string\glossaryentry{\csname glo@#1@index\endcsname
8586 \@gls@encapchar\@glo@numfmt}{\theglsentrycounter}}%
8587 \fi
8588 }
```

\@set@glo@numformat Only had 3 arguments in v2.07

```
8589 \def\@set@glo@numformat#1#2#3{%
8590 \expandafter\@glo@check@mkidxrangear#3\@nil
8591 \protected@edef#1{%
8592 \@glo@prefix setentrycounter []{#2}%
8593 \expandafter\string\csname\@glo@suffix\endcsname
8594 }%
8595 \@gls@checkmkidxchars#1%
8596 }
```

\writeist Redefine \writeist back to the way it was in v2.07, but change \istfile to \glswrite.

```
8597 \ifglxindy
8598 \def\writeist{%
8599 \openout\glswrite=\istfilename
8600 \write\glswrite{;; xindy style file created by the glossaries
8601 package in compatible-2.07 mode}%
8602 \write\glswrite{;; for document '\jobname' on
8603 \the\year-\the\month-\the\day}%
8604 \write\glswrite{^^J; required styles^^J}
8605 \@for\@xdystyle:=\@xdyrequiredstyles\do{%
8606 \ifx\@xdystyle\@empty
8607 \else
8608 \protected@write\glswrite{{(require
8609 \string"\@xdystyle.xdy\string")}}%
8610 \fi
8611 }%
8612 \write\glswrite{^^J%
8613 ; list of allowed attributes (number formats)^^J}%
8614 \write\glswrite{(define-attributes ((\@xdyattributes)))}%
8615 \write\glswrite{^^J; user defined alphabets^^J}%
```

```

8616 \write\glswrite{\@xdyuseralphabets}%
8617 \write\glswrite{^^J; location class definitions^^J}%
8618 \protected@edef\@gls@roman{\@roman{0}\string"
8619 \string"roman-numbers-lowercase\string" :sep \string"}%
8620 \@onelevel@sanitize\@gls@roman
8621 \edef\@tmp{\string" \string"roman-numbers-lowercase\string"
8622 :sep \string}%
8623 \@onelevel@sanitize\@tmp
8624 \ifx\@tmp\@gls@roman
8625 \write\glswrite{(define-location-class
8626 \string"roman-page-numbers\string"^^J\space\space\space
8627 (\string"roman-numbers-lowercase\string")
8628 :min-range-length \@glsminrange)}%
8629 \else
8630 \write\glswrite{(define-location-class
8631 \string"roman-page-numbers\string"^^J\space\space\space
8632 (:sep "\@gls@roman")
8633 :min-range-length \@glsminrange)}%
8634 \fi
8635 \write\glswrite{(define-location-class
8636 \string"Roman-page-numbers\string"^^J\space\space\space
8637 (\string"roman-numbers-uppercase\string")
8638 :min-range-length \@glsminrange)}%
8639 \write\glswrite{(define-location-class
8640 \string"arabic-page-numbers\string"^^J\space\space\space
8641 (\string"arabic-numbers\string")
8642 :min-range-length \@glsminrange)}%
8643 \write\glswrite{(define-location-class
8644 \string"alpha-page-numbers\string"^^J\space\space\space
8645 (\string"alpha\string")
8646 :min-range-length \@glsminrange)}%
8647 \write\glswrite{(define-location-class
8648 \string"Alpha-page-numbers\string"^^J\space\space\space
8649 (\string"ALPHA\string")
8650 :min-range-length \@glsminrange)}%
8651 \write\glswrite{(define-location-class
8652 \string"Appendix-page-numbers\string"^^J\space\space\space
8653 (\string"ALPHA\string"
8654 :sep \string"\@glsAlphacompositor\string"
8655 \string"arabic-numbers\string")
8656 :min-range-length \@glsminrange)}%
8657 \write\glswrite{(define-location-class
8658 \string"arabic-section-numbers\string"^^J\space\space\space
8659 (\string"arabic-numbers\string"
8660 :sep \string"\glscompositor\string"
8661 \string"arabic-numbers\string")
8662 :min-range-length \@glsminrange)}%
8663 \write\glswrite{^^J; user defined location classes}%
8664 \write\glswrite{\@xdyuserlocationdefs}%

```

```

8665 \write\glswrite{^^J; define cross-reference class^^J}%
8666 \write\glswrite{(define-crossref-class \string"see\string"
8667 :unverified )}%
8668 \write\glswrite{(markup-crossref-list
8669 :class \string"see\string"^^J\space\space\space
8670 :open \string"\string\glsseeformat\string"
8671 :close \string"{}\string")}%
8672 \write\glswrite{^^J; define the order of the location classes}%
8673 \write\glswrite{(define-location-class-order
8674 (\@xdylocationclassorder))}%
8675 \write\glswrite{^^J; define the glossary markup^^J}%
8676 \write\glswrite{(markup-index^^J\space\space\space
8677 :open \string"\string
8678 \glossarysection[\string\glossarytoctitle]{\string
8679 \glossarytitle}\string\glossarypreamble\string~n\string\begin
8680 {theglossary}\string\glossaryheader\string~n\string" ^^J\space
8681 \space\space:close \string"\expandafter\@gobble
8682 \string%\string~n\string
8683 \end{theglossary}\string\glossarypostamble
8684 \string~n\string" ^^J\space\space\space
8685 :tree)}%
8686 \write\glswrite{(markup-letter-group-list
8687 :sep \string"\string\glsgroupskip\string~n\string")}%
8688 \write\glswrite{(markup-indexentry
8689 :open \string"\string\relax \string\glsresetentrylist
8690 \string~n\string")}%
8691 \write\glswrite{(markup-locclass-list :open
8692 \string"\glsopenbrace\string\glossaryentrynumbers
8693 \glsopenbrace\string\relax\space \string"^^J\space\space\space
8694 :sep \string", \string"
8695 :close \string"\glsclosebrace\glsclosebrace\string")}%
8696 \write\glswrite{(markup-locref-list
8697 :sep \string"\string\delimN\space\string")}%
8698 \write\glswrite{(markup-range
8699 :sep \string"\string\delimR\space\string")}%
8700 \@onelevel@sanitize\gls@suffixF
8701 \@onelevel@sanitize\gls@suffixFF
8702 \ifx\gls@suffixF\@empty
8703 \else
8704 \write\glswrite{(markup-range
8705 :close "\gls@suffixF" :length 1 :ignore-end)}%
8706 \fi
8707 \ifx\gls@suffixFF\@empty
8708 \else
8709 \write\glswrite{(markup-range
8710 :close "\gls@suffixFF" :length 2 :ignore-end)}%
8711 \fi
8712 \write\glswrite{^^J; define format to use for locations^^J}%
8713 \write\glswrite{\@xdylocref}%

```

```

8714 \write\glswrite{^^J; define letter group list format^^J}%
8715 \write\glswrite{(markup-letter-group-list
8716 :sep \string"\string\glsgroupskip\string~n\string")}%
8717 \write\glswrite{^^J; letter group headings^^J}%
8718 \write\glswrite{(markup-letter-group
8719 :open-head \string"\string\glsgroupheading
8720 \glsoopenbrace\string"^^J\space\space\space
8721 :close-head \string"\glsclosebrace\string")}%
8722 \write\glswrite{^^J; additional letter groups^^J}%
8723 \write\glswrite{\@xdylettergroups}%
8724 \write\glswrite{^^J; additional sort rules^^J}
8725 \write\glswrite{\@xdysortrules}%
8726 \noist}
8727 \else
8728 \edef\@gls@actualchar{\string?}
8729 \edef\@gls@encapchar{\string|}
8730 \edef\@gls@levelchar{\string!}
8731 \edef\@gls@quotechar{\string"}
8732 \def\writeist{\relax
8733 \openout\glswrite=\istfilename
8734 \write\glswrite{\expandafter\@gobble\string\% makeindex style file
8735 created by the glossaries package}
8736 \write\glswrite{\expandafter\@gobble\string\% for document
8737 '\jobname' on \the\year-\the\month-\the\day}
8738 \write\glswrite{actual '\@gls@actualchar'}
8739 \write\glswrite{encap '\@gls@encapchar'}
8740 \write\glswrite{level '\@gls@levelchar'}
8741 \write\glswrite{quote '\@gls@quotechar'}
8742 \write\glswrite{keyword \string"\string\glossaryentry\string"}
8743 \write\glswrite{preamble \string"\string\glossarysection[\string
8744 \glossarytoctitle]{\string\glossarytitle}\string
8745 \glossarypreamble\string\n\string\begin{theglossary}\string
8746 \glossaryheader\string\n\string"}
8747 \write\glswrite{postamble \string"\string%\string\n\string
8748 \end{theglossary}\string\glossarypostamble\string\n
8749 \string"}
8750 \write\glswrite{group_skip \string"\string\glsgroupskip\string\n
8751 \string"}
8752 \write\glswrite{item_0 \string"\string%\string\n\string"}
8753 \write\glswrite{item_1 \string"\string%\string\n\string"}
8754 \write\glswrite{item_2 \string"\string%\string\n\string"}
8755 \write\glswrite{item_01 \string"\string%\string\n\string"}
8756 \write\glswrite{item_x1
8757 \string"\string\relax \string\glsresetentrylist\string\n
8758 \string"}
8759 \write\glswrite{item_12 \string"\string%\string\n\string"}
8760 \write\glswrite{item_x2
8761 \string"\string\relax \string\glsresetentrylist\string\n
8762 \string"}

```

```

8763 \write\glswrite{delim_0 \string"\string\{\string
8764 \glossaryentrynumbers\string\{\string\relax \string"}
8765 \write\glswrite{delim_1 \string"\string\{\string
8766 \glossaryentrynumbers\string\{\string\relax \string"}
8767 \write\glswrite{delim_2 \string"\string\{\string
8768 \glossaryentrynumbers\string\{\string\relax \string"}
8769 \write\glswrite{delim_t \string"\string\}\string\}\string"}
8770 \write\glswrite{delim_n \string"\string\delimN \string"}
8771 \write\glswrite{delim_r \string"\string\delimR \string"}
8772 \write\glswrite{headings_flag 1}
8773 \write\glswrite{heading_prefix
8774 \string"\string\glsgroupheading\string\{\string"}
8775 \write\glswrite{heading_suffix
8776 \string"\string\}\string\relax
8777 \string\glsresetentrylist \string"}
8778 \write\glswrite{symhead_positive \string"glssymbols\string"}
8779 \write\glswrite{numhead_positive \string"glnumbers\string"}
8780 \write\glswrite{page_compositor \string"glscpositor\string"}
8781 \@gls@escbsdq\gls@suffixF
8782 \@gls@escbsdq\gls@suffixFF
8783 \ifx\gls@suffixF\@empty
8784 \else
8785 \write\glswrite{suffix_2p \string"\gls@suffixF\string"}
8786 \fi
8787 \ifx\gls@suffixFF\@empty
8788 \else
8789 \write\glswrite{suffix_3p \string"\gls@suffixFF\string"}
8790 \fi
8791 \noist
8792 }
8793 \fi

```

\noist

```
8794 \renewcommand*{\noist}{\let\writeist\relax}
```

Compatibility macros.

```

8795 \NeedsTeXFormat{LaTeX2e}
8796 \ProvidesPackage{glossaries-compatible-307}[2013/11/14 v4.0 (NLCT)]

```

Compatibility macros for predefined glossary styles:

`compatglossarystyle` Defines a compatibility glossary style.

```

8797 \newcommand{\compatglossarystyle}[2]{%
8798 \ifcsundef{@glscompstyle@#1}%
8799 {%
8800 \csdef{@glscompstyle@#1}{#2}%
8801 }%
8802 {%
8803 \PackageError{glossaries}{Glossary compatibility style ‘#1’ is already defined}{}%
8804 }%

```


8805 }

Backward compatible inline style.

```
8806 \compatglossarystyle{inline}{%
8807   \renewcommand{\glossaryentryfield}[5]{%
8808     \glsinlinedopostchild
8809     \gls@inlinesep
8810     \def\glo@desc{##3}%
8811     \def\@no@post@desc{\nopostdesc}%
8812     \glsentryitem{##1}\glsinlinenameformat{##1}{##2}%
8813     \ifx\glo@desc\@no@post@desc
8814       \glsinlineemptydescformat{##4}{##5}%
8815     \else
8816       \ifstrempy{##3}%
8817         {\glsinlineemptydescformat{##4}{##5}}%
8818         {\glsinlinedescformat{##3}{##4}{##5}}%
8819       \fi
8820     \ifglshaschildren{##1}%
8821     {%
8822       \glsresetsubentrycounter
8823       \glsinlineparentchildseparator
8824       \def\gls@inlinesubsep{}%
8825       \def\gls@inlinepostchild{\glsinlinepostchild}%
8826     }%
8827     {}%
8828     \def\gls@inlinesep{\glsinlineseparator}%
8829   }%
```

Sub-entries display description:

```
8830 \renewcommand{\glossarysubentryfield}[6]{%
8831   \gls@inlinesubsep%
8832   \glsinlinesubnameformat{##2}{##3}%
8833   \glssubentryitem{##2}\glsinlinesubdescformat{##4}{##5}{##6}%
8834   \def\gls@inlinesubsep{\glsinlinesubseparator}%
8835 }%
8836 }
```

Backward compatible list style.

```
8837 \compatglossarystyle{list}{%
8838   \renewcommand*\glossaryentryfield[5]{%
8839     \item[\glsentryitem{##1}\glstarget{##1}{##2}]
8840     ##3\glspostdescription\space ##5}%

```

Sub-entries continue on the same line:

```
8841 \renewcommand*\glossarysubentryfield[6]{%
8842   \glssubentryitem{##2}%
8843   \glstarget{##2}{\strut}##4\glspostdescription\space ##6.}%
8844 }
```

Backward compatible listgroup style.

```
8845 \compatglossarystyle{listgroup}{%
```

```
8846 \csuse{@glscompstyle@list}%
8847 }%
```

Backward compatible listhypergroup style.

```
8848 \compatglossarystyle{listhypergroup}{%
8849 \csuse{@glscompstyle@list}%
8850 }%
```

Backward compatible altlist style.

```
8851 \compatglossarystyle{altlist}{%
8852 \renewcommand*{\glossaryentryfield}[5]{%
8853 \item[\glsentryitem{##1}\glstarget{##1}{##2}]%
8854 \mbox{}\par\nobreak\@afterheading
8855 ##3\glspostdescription\space ##5}%
8856 \renewcommand{\glossarysubentryfield}[6]{%
8857 \par
8858 \glssubentryitem{##2}%
8859 \glstarget{##2}{\strut}##4\glspostdescription\space ##6}%
8860 }%
```

Backward compatible altlistgroup style.

```
8861 \compatglossarystyle{altlistgroup}{%
8862 \csuse{@glscompstyle@altlist}%
8863 }%
```

Backward compatible altlisthypergroup style.

```
8864 \compatglossarystyle{altlisthypergroup}{%
8865 \csuse{@glscompstyle@altlist}%
8866 }%
```

Backward compatible listdotted style.

```
8867 \compatglossarystyle{listdotted}{%
8868 \renewcommand*{\glossaryentryfield}[5]{%
8869 \item[]\makebox[\glslistdottedwidth][1]{%
8870 \glsentryitem{##1}\glstarget{##1}{##2}%
8871 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##3}%
8872 \renewcommand*{\glossarysubentryfield}[6]{%
8873 \item[]\makebox[\glslistdottedwidth][1]{%
8874 \glssubentryitem{##2}%
8875 \glstarget{##2}{##3}%
8876 \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}##4}%
8877 }%
```

Backward compatible sublistdotted style.

```
8878 \compatglossarystyle{sublistdotted}{%
8879 \csuse{@glscompstyle@listdotted}%
8880 \renewcommand*{\glossaryentryfield}[5]{%
8881 \item[\glsentryitem{##1}\glstarget{##1}{##2}]}%
8882 }%
```

Backward compatible long style.

```
8883 \compatglossarystyle{long}{%
```

```

8884 \renewcommand*\glossaryentryfield}[5]{%
8885   \glstryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
8886 \renewcommand*\glossarysubentryfield}[6]{%
8887   &
8888   \glssubentryitem{##2}%
8889   \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
8890 }%

```

Backward compatible longborder style.

```

8891 \compatglossarystyle{longborder}{%
8892 \csuse{@glscmpstyle@long}%
8893 }%

```

Backward compatible longheader style.

```

8894 \compatglossarystyle{longheader}{%
8895 \csuse{@glscmpstyle@long}%
8896 }%

```

Backward compatible longheaderborder style.

```

8897 \compatglossarystyle{longheaderborder}{%
8898 \csuse{@glscmpstyle@long}%
8899 }%

```

Backward compatible long3col style.

```

8900 \compatglossarystyle{long3col}{%
8901 \renewcommand*\glossaryentryfield}[5]{%
8902   \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
8903 \renewcommand*\glossarysubentryfield}[6]{%
8904   &
8905   \glssubentryitem{##2}%
8906   \glstarget{##2}{\strut}##4 & ##6\\}%
8907 }%

```

Backward compatible long3colborder style.

```

8908 \compatglossarystyle{long3colborder}{%
8909 \csuse{@glscmpstyle@long3col}%
8910 }%

```

Backward compatible long3colheader style.

```

8911 \compatglossarystyle{long3colheader}{%
8912 \csuse{@glscmpstyle@long3col}%
8913 }%

```

Backward compatible long3colheaderborder style.

```

8914 \compatglossarystyle{long3colheaderborder}{%
8915 \csuse{@glscmpstyle@long3col}%
8916 }%

```

Backward compatible long4col style.

```

8917 \compatglossarystyle{long4col}{%
8918 \renewcommand*\glossaryentryfield}[5]{%
8919   \glstryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%

```

```

8920 \renewcommand*{\glossarysubentryfield}[6]{%
8921     &
8922     \glssubentryitem{##2}%
8923     \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
8924 }%

```

Backward compatible long4colheader style.

```

8925 \compatglossarystyle{long4colheader}{%
8926 \csuse{@glscompstyle@long4col}%
8927 }%

```

Backward compatible long4colborder style.

```

8928 \compatglossarystyle{long4colborder}{%
8929 \csuse{@glscompstyle@long4col}%
8930 }%

```

Backward compatible long4colheaderborder style.

```

8931 \compatglossarystyle{long4colheaderborder}{%
8932 \csuse{@glscompstyle@long4col}%
8933 }%

```

Backward compatible altlong4col style.

```

8934 \compatglossarystyle{altlong4col}{%
8935 \csuse{@glscompstyle@long4col}%
8936 }%

```

Backward compatible altlong4colheader style.

```

8937 \compatglossarystyle{altlong4colheader}{%
8938 \csuse{@glscompstyle@long4col}%
8939 }%

```

Backward compatible altlong4colborder style.

```

8940 \compatglossarystyle{altlong4colborder}{%
8941 \csuse{@glscompstyle@long4col}%
8942 }%

```

Backward compatible altlong4colheaderborder style.

```

8943 \compatglossarystyle{altlong4colheaderborder}{%
8944 \csuse{@glscompstyle@long4col}%
8945 }%

```

Backward compatible long style.

```

8946 \compatglossarystyle{longragged}{%
8947 \renewcommand*{\glossaryentryfield}[5]{%
8948     \glstarget{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
8949     \tabularnewline}%
8950 \renewcommand*{\glossarysubentryfield}[6]{%
8951     &
8952     \glssubentryitem{##2}%
8953     \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
8954     \tabularnewline}%
8955 }%

```

Backward compatible longraggedborder style.

```
8956 \compatglossarystyle{longraggedborder}{%
8957 \csuse{@glscompstyle@longragged}%
8958 }%
```

Backward compatible longraggedheader style.

```
8959 \compatglossarystyle{longraggedheader}{%
8960 \csuse{@glscompstyle@longragged}%
8961 }%
```

Backward compatible longraggedheaderborder style.

```
8962 \compatglossarystyle{longraggedheaderborder}{%
8963 \csuse{@glscompstyle@longragged}%
8964 }%
```

Backward compatible longragged3col style.

```
8965 \compatglossarystyle{longragged3col}{%
8966 \renewcommand*{\glossaryentryfield}[5]{%
8967 \glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
8968 \renewcommand*{\glossarysubentryfield}[6]{%
8969 &
8970 \glssubentryitem{##2}%
8971 \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
8972 }%
```

Backward compatible longragged3colborder style.

```
8973 \compatglossarystyle{longragged3colborder}{%
8974 \csuse{@glscompstyle@longragged3col}%
8975 }%
```

Backward compatible longragged3colheader style.

```
8976 \compatglossarystyle{longragged3colheader}{%
8977 \csuse{@glscompstyle@longragged3col}%
8978 }%
```

Backward compatible longragged3colheaderborder style.

```
8979 \compatglossarystyle{longragged3colheaderborder}{%
8980 \csuse{@glscompstyle@longragged3col}%
8981 }%
```

Backward compatible altlongragged4col style.

```
8982 \compatglossarystyle{altlongragged4col}{%
8983 \renewcommand*{\glossaryentryfield}[5]{%
8984 \glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
8985 \renewcommand*{\glossarysubentryfield}[6]{%
8986 &
8987 \glssubentryitem{##2}%
8988 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
8989 }%
```

Backward compatible altlongragged4colheader style.

```
8990 \compatglossarystyle{altlongragged4colheader}{%
```

```
8991 \csuse{@glscompstyle@altlong4col}%
8992 }%
```

Backward compatible altlongragged4colborder style.

```
8993 \compatglossarystyle{altlongragged4colborder}{%
8994 \csuse{@glscompstyle@altlong4col}%
8995 }%
```

Backward compatible altlongragged4colheaderborder style.

```
8996 \compatglossarystyle{altlongragged4colheaderborder}{%
8997 \csuse{@glscompstyle@altlong4col}%
8998 }%
```

Backward compatible index style.

```
8999 \compatglossarystyle{index}{%
9000 \renewcommand*{\glossaryentryfield}[5]{%
9001 \item\glstarget{##1}{##2}}%
9002 \ifx\relax##4\relax
9003 \else
9004 \space{##4}%
9005 \fi
9006 \space ##3\glspostdescription \space ##5}%
9007 \renewcommand*{\glossarysubentryfield}[6]{%
9008 \ifcase##1\relax
9009 % level 0
9010 \item
9011 \or
9012 % level 1
9013 \subitem
9014 \glssubentryitem{##2}%
9015 \else
9016 % all other levels
9017 \subsubitem
9018 \fi
9019 \textbf{\glstarget{##2}{##3}}%
9020 \ifx\relax##5\relax
9021 \else
9022 \space{##5}%
9023 \fi
9024 \space##4\glspostdescription\space ##6}%
9025 }%
```

Backward compatible indexgroup style.

```
9026 \compatglossarystyle{indexgroup}{%
9027 \csuse{@glscompstyle@index}%
9028 }%
```

Backward compatible indexhypergroup style.

```
9029 \compatglossarystyle{indexhypergroup}{%
9030 \csuse{@glscompstyle@index}%
9031 }%
```

Backward compatible tree style.

```
9032 \compatglossarystyle{tree}{%
9033   \renewcommand{\glossaryentryfield}[5]{%
9034     \hangindentOpt\relax
9035     \parindentOpt\relax
9036     \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
9037     \ifx\relax##4\relax
9038     \else
9039       \space{##4}%
9040     \fi
9041     \space ##3\glspostdescription \space ##5\par}%
9042 \renewcommand{\glossarysubentryfield}[6]{%
9043   \hangindent##1\glstreeindent\relax
9044   \parindent##1\glstreeindent\relax
9045   \ifnum##1=1\relax
9046     \glssubentryitem{##2}%
9047   \fi
9048   \textbf{\glstarget{##2}{##3}}%
9049   \ifx\relax##5\relax
9050   \else
9051     \space{##5}%
9052   \fi
9053   \space##4\glspostdescription\space ##6\par}%
9054 }
```

Backward compatible treegroup style.

```
9055 \compatglossarystyle{treegroup}{%
9056   \csuse{@glscmpstyle@tree}%
9057 }
```

Backward compatible treehypergroup style.

```
9058 \compatglossarystyle{treehypergroup}{%
9059   \csuse{@glscmpstyle@tree}%
9060 }
```

Backward compatible treenoname style.

```
9061 \compatglossarystyle{treenoname}{%
9062   \renewcommand{\glossaryentryfield}[5]{%
9063     \hangindentOpt\relax
9064     \parindentOpt\relax
9065     \glstentryitem{##1}\textbf{\glstarget{##1}{##2}}%
9066     \ifx\relax##4\relax
9067     \else
9068       \space{##4}%
9069     \fi
9070     \space ##3\glspostdescription \space ##5\par}%
9071 \renewcommand{\glossarysubentryfield}[6]{%
9072   \hangindent##1\glstreeindent\relax
9073   \parindent##1\glstreeindent\relax
9074   \ifnum##1=1\relax
```

```

9075     \glssubentryitem{##2}%
9076     \fi
9077     \glstarget{##2}{\strut}%
9078     ##4\glspostdescription\space ##6\par}%
9079 }%

```

Backward compatible treenonamegroup style.

```

9080 \compatglossarystyle{treenonamegroup}{%
9081   \csuse{@glscompstyle@treenoname}%
9082 }%

```

Backward compatible treenonamehypergroup style.

```

9083 \compatglossarystyle{treenonamehypergroup}{%
9084   \csuse{@glscompstyle@treenoname}%
9085 }%

```

Backward compatible altree style.

```

9086 \compatglossarystyle{almtree}{%
9087   \renewcommand{\glossaryentryfield}[5]{%
9088     \ifnum \@gls@prevlevel=0\relax
9089     \else
9090       \settowidth{\glstreeindent}{\textbf{\@glswidestname\space}}%
9091       \hangindent\glstreeindent
9092       \parindent\glstreeindent
9093     \fi
9094     \makebox[0pt][r]{\makebox[\glstreeindent][l]{%
9095       \glssubentryitem{##1}\textbf{\glstarget{##1}{##2}}}}%
9096     \ifx\relax##4\relax
9097     \else
9098       (##4)\space
9099     \fi
9100     ##3\glspostdescription \space ##5\par
9101     \def\@gls@prevlevel{0}%
9102   }%
9103   \renewcommand{\glossarysubentryfield}[6]{%
9104     \ifnum##1=1\relax
9105       \glssubentryitem{##2}%
9106     \fi
9107     \ifnum \@gls@prevlevel=##1\relax
9108     \else
9109       \@ifundefined{@glswidestname\romannumeral##1}{%
9110         \settowidth{\gls@tmplen}{\textbf{\@glswidestname\space}}{%
9111         \settowidth{\gls@tmplen}{\textbf{%
9112           \csname @glswidestname\romannumeral##1\endcsname\space}}}%
9113       \ifnum \@gls@prevlevel<##1\relax
9114         \setlength\glstreeindent\gls@tmplen
9115         \addtolength\glstreeindent\parindent
9116         \parindent\glstreeindent
9117       \else
9118         \@ifundefined{@glswidestname\romannumeral\@gls@prevlevel}{%
9119         \settowidth{\glstreeindent}{\textbf{%

```



```

9120         \@glswidestname\space}}}{%
9121         \settowidth{\glstreeindent}{\textbf{%
9122         \csname @glswidestname\romannumeral\@gls@prevlevel
9123         \endcsname\space}}}{%
9124         \addtolength\parindent{-\glstreeindent}%
9125         \setlength\glstreeindent\parindent
9126     \fi
9127 \fi
9128 \hangindent\glstreeindent
9129 \makebox[0pt][r]{\makebox[\@gls@tmplen][l]{%
9130     \textbf{\glstarget{##2}{##3}}}}}{%
9131 \ifx##5\relax\relax
9132 \else
9133     (##5)\space
9134 \fi
9135 ##4\glspostdescription\space ##6\par
9136 \def\@gls@prevlevel{##1}}}{%
9137 }%
9138 }%

```

Backward compatible alttreegroup style.

```

9139 \compatglossarystyle{alttreegroup}{%
9140 \csuse{@glscompstyle@almtree}}%
9141 }%

```

Backward compatible alttreehypergroup style.

```

9142 \compatglossarystyle{alttreehypergroup}{%
9143 \csuse{@glscompstyle@almtree}}%
9144 }%

```

Backward compatible mcolindex style.

```

9145 \compatglossarystyle{mcolindex}{%
9146 \csuse{@glscompstyle@index}}%
9147 }%

```

Backward compatible mcolindexgroup style.

```

9148 \compatglossarystyle{mcolindexgroup}{%
9149 \csuse{@glscompstyle@index}}%
9150 }%

```

Backward compatible mcolindexhypergroup style.

```

9151 \compatglossarystyle{mcolindexhypergroup}{%
9152 \csuse{@glscompstyle@index}}%
9153 }%

```

Backward compatible mcoltree style.

```

9154 \compatglossarystyle{mcoltree}{%
9155 \csuse{@glscompstyle@tree}}%
9156 }%

```

Backward compatible mcoltreegroup style.

```

9157 \compatglossarystyle{mcolindextreegroup}{%

```

```

9158 \csuse{@glscompstyle@tree}%
9159 }%

  Backward compatible mcoltreehypergroup style.
9160 \compatglossarystyle{mcolindextreehypergroup}{%
9161 \csuse{@glscompstyle@tree}%
9162 }%

  Backward compatible mcoltreenoname style.
9163 \compatglossarystyle{mcoltreenoname}{%
9164 \csuse{@glscompstyle@tree}%
9165 }%

  Backward compatible mcoltreenonamegroup style.
9166 \compatglossarystyle{mcoltreenonamegroup}{%
9167 \csuse{@glscompstyle@tree}%
9168 }%

  Backward compatible mcoltreenonamehypergroup style.
9169 \compatglossarystyle{mcoltreenonamehypergroup}{%
9170 \csuse{@glscompstyle@tree}%
9171 }%

  Backward compatible mcolalmtree style.
9172 \compatglossarystyle{mcolalmtree}{%
9173 \csuse{@glscompstyle@almtree}%
9174 }%

  Backward compatible mcolalmtreegroup style.
9175 \compatglossarystyle{mcolalmtreegroup}{%
9176 \csuse{@glscompstyle@almtree}%
9177 }%

  Backward compatible mcolalmtreehypergroup style.
9178 \compatglossarystyle{mcolalmtreehypergroup}{%
9179 \csuse{@glscompstyle@almtree}%
9180 }%

  Backward compatible superragged style.
9181 \compatglossarystyle{superragged}{%
9182 \renewcommand*{\glossaryentryfield}[5]{%
9183 \glsentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5%
9184 \tabularnewline}%
9185 \renewcommand*{\glossarysubentryfield}[6]{%
9186 &
9187 \glssubentryitem{##2}%
9188 \glstarget{##2}{\strut}##4\glspostdescription\space ##6%
9189 \tabularnewline}%
9190 }%

  Backward compatible superraggedborder style.
9191 \compatglossarystyle{superraggedborder}{%
9192 \csuse{@glscompstyle@superragged}%
9193 }%

```

Backward compatible superraggedheader style.

```
9194 \compatglossarystyle{superraggedheader}{%
9195 \csuse{@glscompstyle@superragged}%
9196 }%
```

Backward compatible superraggedheaderborder style.

```
9197 \compatglossarystyle{superraggedheaderborder}{%
9198 \csuse{@glscompstyle@superragged}%
9199 }%
```

Backward compatible superragged3col style.

```
9200 \compatglossarystyle{superragged3col}{%
9201 \renewcommand*{\glossaryentryfield}[5]{%
9202 \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\tabularnewline}%
9203 \renewcommand*{\glossarysubentryfield}[6]{%
9204 &
9205 \glssubentryitem{##2}%
9206 \glstarget{##2}{\strut}##4 & ##6\tabularnewline}%
9207 }%
```

Backward compatible superragged3colborder style.

```
9208 \compatglossarystyle{superragged3colborder}{%
9209 \csuse{@glscompstyle@superragged3col}%
9210 }%
```

Backward compatible superragged3colheader style.

```
9211 \compatglossarystyle{superragged3colheader}{%
9212 \csuse{@glscompstyle@superragged3col}%
9213 }%
```

Backward compatible superragged3colheaderborder style.

```
9214 \compatglossarystyle{superragged3colheaderborder}{%
9215 \csuse{@glscompstyle@superragged3col}%
9216 }%
```

Backward compatible altsuperragged4col style.

```
9217 \compatglossarystyle{altsuperragged4col}{%
9218 \renewcommand*{\glossaryentryfield}[5]{%
9219 \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\tabularnewline}%
9220 \renewcommand*{\glossarysubentryfield}[6]{%
9221 &
9222 \glssubentryitem{##2}%
9223 \glstarget{##2}{\strut}##4 & ##5 & ##6\tabularnewline}%
9224 }%
```

Backward compatible altsuperragged4colheader style.

```
9225 \compatglossarystyle{altsuperragged4colheader}{%
9226 \csuse{@glscompstyle@altsuperragged4col}%
9227 }%
```

Backward compatible altsuperragged4colborder style.

```
9228 \compatglossarystyle{altsuperragged4colborder}{%
```

9229 \csuse{@glscompstyle@altsuperragged4col}%
9230 }%

Backward compatible altsuperragged4colheaderborder style.

9231 \compatglossarystyle{altsuperragged4colheaderborder}{%
9232 \csuse{@glscompstyle@altsuperragged4col}%
9233 }%

Backward compatible super style.

9234 \compatglossarystyle{super}{%
9235 \renewcommand*{\glossaryentryfield}[5]{%
9236 \glsentryitem{##1}\glstarget{##1}{##2} & ##3\glspostdescription\space ##5\\}%
9237 \renewcommand*{\glossarysubentryfield}[6]{%
9238 &
9239 \glssubentryitem{##2}%
9240 \glstarget{##2}{\strut}##4\glspostdescription\space ##6\\}%
9241 }%

Backward compatible superborder style.

9242 \compatglossarystyle{superborder}{%
9243 \csuse{@glscompstyle@super}%
9244 }%

Backward compatible superheader style.

9245 \compatglossarystyle{superheader}{%
9246 \csuse{@glscompstyle@super}%
9247 }%

Backward compatible superheaderborder style.

9248 \compatglossarystyle{superheaderborder}{%
9249 \csuse{@glscompstyle@super}%
9250 }%

Backward compatible super3col style.

9251 \compatglossarystyle{super3col}{%
9252 \renewcommand*{\glossaryentryfield}[5]{%
9253 \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##5\\}%
9254 \renewcommand*{\glossarysubentryfield}[6]{%
9255 &
9256 \glssubentryitem{##2}%
9257 \glstarget{##2}{\strut}##4 & ##6\\}%
9258 }%

Backward compatible super3colborder style.

9259 \compatglossarystyle{super3colborder}{%
9260 \csuse{@glscompstyle@super3col}%
9261 }%

Backward compatible super3colheader style.

9262 \compatglossarystyle{super3colheader}{%
9263 \csuse{@glscompstyle@super3col}%
9264 }%

Backward compatible super3colheaderborder style.

```
9265 \compatglossarystyle{super3colheaderborder}{%
9266 \csuse{@glscompstyle@super3col}%
9267 }%
```

Backward compatible super4col style.

```
9268 \compatglossarystyle{super4col}{%
9269 \renewcommand*{\glossaryentryfield}[5]{%
9270 \glsentryitem{##1}\glstarget{##1}{##2} & ##3 & ##4 & ##5\\}%
9271 \renewcommand*{\glossarysubentryfield}[6]{%
9272 &
9273 \glssubentryitem{##2}%
9274 \glstarget{##2}{\strut}##4 & ##5 & ##6\\}%
9275 }%
```

Backward compatible super4colheader style.

```
9276 \compatglossarystyle{super4colheader}{%
9277 \csuse{@glscompstyle@super4col}%
9278 }%
```

Backward compatible super4colborder style.

```
9279 \compatglossarystyle{super4colborder}{%
9280 \csuse{@glscompstyle@super4col}%
9281 }%
```

Backward compatible super4colheaderborder style.

```
9282 \compatglossarystyle{super4colheaderborder}{%
9283 \csuse{@glscompstyle@super4col}%
9284 }%
```

Backward compatible altsuper4col style.

```
9285 \compatglossarystyle{altsuper4col}{%
9286 \csuse{@glscompstyle@super4col}%
9287 }%
```

Backward compatible altsuper4colheader style.

```
9288 \compatglossarystyle{altsuper4colheader}{%
9289 \csuse{@glscompstyle@super4col}%
9290 }%
```

Backward compatible altsuper4colborder style.

```
9291 \compatglossarystyle{altsuper4colborder}{%
9292 \csuse{@glscompstyle@super4col}%
9293 }%
```

Backward compatible altsuper4colheaderborder style.

```
9294 \compatglossarystyle{altsuper4colheaderborder}{%
9295 \csuse{@glscompstyle@super4col}%
9296 }%
```

7 Accessibility Support (glossaries-accsupp Code)

The package is experimental. It is intended to provide a means of using the PDF accessibility support in glossary entries. See the documentation for further details about accessibility support.

```
9297 \NeedsTeXFormat{LaTeX2e}
```

Package version number now in line with main glossaries package number but will only be updated when glossaries-accsupp.sty is modified.

```
9298 \ProvidesPackage{glossaries-accsupp}[2014/07/30 v4.08 (NLCT)
```

```
9299 Experimental glossaries accessibility]
```

Pass all options to glossaries:

```
9300 \DeclareOption*{\PassOptionsToPackage{\CurrentOption}{glossaries}}
```

Process options:

```
9301 \ProcessOptions
```

compatibleglossentry Override style compatibility macros:

```
9302 \def\compatibleglossentry#1#2{%
```

```
9303 \toks@{#2}%
```

```
9304 \protected@edef\@do@glossentry{%
```

```
9305 \noexpand\accsuppglossaryentryfield{#1}%
```

```
9306 {\noexpand\glsnamefont
```

```
9307 {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@name\endcsname}}%
```

```
9308 {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@desc\endcsname}}%
```

```
9309 {\expandafter\expandonce\csname glo@\glsdetoklabel{#1}@symbol\endcsname}}%
```

```
9310 {\the\toks@}%
```

```
9311 }%
```

```
9312 \@do@glossentry
```

```
9313 }
```

compatiblesubglossentry

```
9314 \def\compatiblesubglossentry#1#2#3{%
```

```
9315 \toks@{#3}%
```

```
9316 \protected@edef\@do@subglossentry{%
```

```
9317 \noexpand\accsuppglossarysubentryfield{\number#1}%
```

```
9318 {#2}%
```

```
9319 {\noexpand\glsnamefont
```

```
9320 {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@name\endcsname}}%
```

```
9321 {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@desc\endcsname}}%
```

```
9322 {\expandafter\expandonce\csname glo@\glsdetoklabel{#2}@symbol\endcsname}}%
```

```
9323 {\the\toks@}%
```

```
9324 }%
```

```
9325 \@do@subglossentry
```

```
9326 }
```

Required packages:

```
9327 \RequirePackage{glossaries}
```

```
9328 \RequirePackage{accsupp}
```

7.1 Defining Replacement Text

The version 0.1 stored the replacement text in the symbol key. This has been changed to use the new keys defined here. Example of use:

```
\newglossaryentry{dr}{name=Dr,description={},access={Doctor}}
```

access The replacement text corresponding to the name key:

```
9329 \define@key{glossentry}{access}{%  
9330   \def\@glo@access{#1}%  
9331 }
```

textaccess The replacement text corresponding to the text key:

```
9332 \define@key{glossentry}{textaccess}{%  
9333   \def\@glo@textaccess{#1}%  
9334 }
```

firstaccess The replacement text corresponding to the first key:

```
9335 \define@key{glossentry}{firstaccess}{%  
9336   \def\@glo@firstaccess{#1}%  
9337 }
```

pluralaccess The replacement text corresponding to the plural key:

```
9338 \define@key{glossentry}{pluralaccess}{%  
9339   \def\@glo@pluralaccess{#1}%  
9340 }
```

firstpluralaccess The replacement text corresponding to the firstplural key:

```
9341 \define@key{glossentry}{firstpluralaccess}{%  
9342   \def\@glo@firstpluralaccess{#1}%  
9343 }
```

symbolaccess The replacement text corresponding to the symbol key:

```
9344 \define@key{glossentry}{symbolaccess}{%  
9345   \def\@glo@symbolaccess{#1}%  
9346 }
```

symbolpluralaccess The replacement text corresponding to the symbolplural key:

```
9347 \define@key{glossentry}{symbolpluralaccess}{%  
9348   \def\@glo@symbolpluralaccess{#1}%  
9349 }
```

descriptionaccess The replacement text corresponding to the description key:

```
9350 \define@key{glossentry}{descriptionaccess}{%  
9351   \def\@glo@descaccess{#1}%  
9352 }
```

descriptionpluralaccess The replacement text corresponding to the descriptionplural key:

```
9353 \define@key{glossentry}{descriptionpluralaccess}{%
9354   \def\@glo@descpluralaccess{#1}%
9355 }
```

shortaccess The replacement text corresponding to the short key:

```
9356 \define@key{glossentry}{shortaccess}{%
9357   \def\@glo@shortaccess{#1}%
9358 }
```

shortpluralaccess The replacement text corresponding to the shortplural key:

```
9359 \define@key{glossentry}{shortpluralaccess}{%
9360   \def\@glo@shortpluralaccess{#1}%
9361 }
```

longaccess The replacement text corresponding to the long key:

```
9362 \define@key{glossentry}{longaccess}{%
9363   \def\@glo@longaccess{#1}%
9364 }
```

longpluralaccess The replacement text corresponding to the longplural key:

```
9365 \define@key{glossentry}{longpluralaccess}{%
9366   \def\@glo@longpluralaccess{#1}%
9367 }
```

There are no equivalent keys for the user1...user6 keys. The replacement text would have to be explicitly put in the value, e.g., user1={\glsaccsupp{inches}{in}}.

Append these new keys to \@gls@keymap:

```
9368 \appto\@gls@keymap{,%
9369   {access}{access},%
9370   {textaccess}{textaccess},%
9371   {firstaccess}{firstaccess},%
9372   {pluralaccess}{pluralaccess},%
9373   {firstpluralaccess}{firstpluralaccess},%
9374   {symbolaccess}{symbolaccess},%
9375   {symbolpluralaccess}{symbolpluralaccess},%
9376   {descaccess}{descaccess},%
9377   {descpluralaccess}{descpluralaccess},%
9378   {shortaccess}{shortaccess},%
9379   {shortpluralaccess}{shortpluralaccess},%
9380   {longaccess}{longaccess},%
9381   {longpluralaccess}{longpluralaccess}%
9382 }
```

\@gls@noaccess Indicates that no replacement text has been provided.

```
9383 \def\@gls@noaccess{\relax}
```


Add to the start hook (the access key is initialised to the value of the symbol key at the start for backwards compatibility):

```
9384 \let\@gls@oldnewglossaryentryprehook\@newglossaryentryprehook
9385 \renewcommand*\@newglossaryentryprehook}{%
9386   \@gls@oldnewglossaryentryprehook
9387   \def\@glo@access{\@glo@symbol}%
```

Initialise the other keys:

```
9388   \def\@glo@textaccess{\@glo@access}%
9389   \def\@glo@firstaccess{\@glo@access}%
9390   \def\@glo@pluralaccess{\@glo@textaccess}%
9391   \def\@glo@firstpluralaccess{\@glo@pluralaccess}%
9392   \def\@glo@symbolaccess{\relax}%
9393   \def\@glo@symbolpluralaccess{\@glo@symbolaccess}%
9394   \def\@glo@descaccess{\relax}%
9395   \def\@glo@descpluralaccess{\@glo@descaccess}%
9396   \def\@glo@shortaccess{\relax}%
9397   \def\@glo@shortpluralaccess{\@glo@shortaccess}%
9398   \def\@glo@longaccess{\relax}%
9399   \def\@glo@longpluralaccess{\@glo@longaccess}%
9400 }
```

Add to the end hook:

```
9401 \let\@gls@oldnewglossaryentryposthook\@newglossaryentryposthook
9402 \renewcommand*\@newglossaryentryposthook}{%
9403   \@gls@oldnewglossaryentryposthook
```

Store the access information:

```
9404   \expandafter
9405     \protected@xdef\csname glo@\@glo@label @access\endcsname{%
9406     \@glo@access}%
9407   \expandafter
9408     \protected@xdef\csname glo@\@glo@label @textaccess\endcsname{%
9409     \@glo@textaccess}%
9410   \expandafter
9411     \protected@xdef\csname glo@\@glo@label @firstaccess\endcsname{%
9412     \@glo@firstaccess}%
9413   \expandafter
9414     \protected@xdef\csname glo@\@glo@label @pluralaccess\endcsname{%
9415     \@glo@pluralaccess}%
9416   \expandafter
9417     \protected@xdef\csname glo@\@glo@label @firstpluralaccess\endcsname{%
9418     \@glo@firstpluralaccess}%
9419   \expandafter
9420     \protected@xdef\csname glo@\@glo@label @symbolaccess\endcsname{%
9421     \@glo@symbolaccess}%
9422   \expandafter
9423     \protected@xdef\csname glo@\@glo@label @symbolpluralaccess\endcsname{%
9424     \@glo@symbolpluralaccess}%
9425   \expandafter
```

```

9426   \protected@xdef\csname glo@\@glo@label @descaccess\endcsname{%
9427     \@glo@descaccess}%
9428   \expandafter
9429   \protected@xdef\csname glo@\@glo@label @descpluralaccess\endcsname{%
9430     \@glo@descpluralaccess}%
9431   \expandafter
9432   \protected@xdef\csname glo@\@glo@label @shortaccess\endcsname{%
9433     \@glo@shortaccess}%
9434   \expandafter
9435   \protected@xdef\csname glo@\@glo@label @shortpluralaccess\endcsname{%
9436     \@glo@shortpluralaccess}%
9437   \expandafter
9438   \protected@xdef\csname glo@\@glo@label @longaccess\endcsname{%
9439     \@glo@longaccess}%
9440   \expandafter
9441   \protected@xdef\csname glo@\@glo@label @longpluralaccess\endcsname{%
9442     \@glo@longpluralaccess}%
9443 }

```

7.2 Accessing Replacement Text

`\glsentryaccess` Get the value of the access key for the entry with the given label:

```

9444 \newcommand*\glsentryaccess[1]{%
9445   \@gls@entry@field{#1}{access}%
9446 }

```

`\glsentrytextaccess` Get the value of the textaccess key for the entry with the given label:

```

9447 \newcommand*\glsentrytextaccess[1]{%
9448   \@gls@entry@field{#1}{textaccess}%
9449 }

```

`\glsentryfirstaccess` Get the value of the firstaccess key for the entry with the given label:

```

9450 \newcommand*\glsentryfirstaccess[1]{%
9451   \@gls@entry@field{#1}{firstaccess}%
9452 }

```

`\glsentrypluralaccess` Get the value of the pluralaccess key for the entry with the given label:

```

9453 \newcommand*\glsentrypluralaccess[1]{%
9454   \@gls@entry@field{#1}{pluralaccess}%
9455 }

```

`\glsentryfirstpluralaccess` Get the value of the firstpluralaccess key for the entry with the given label:

```

9456 \newcommand*\glsentryfirstpluralaccess[1]{%
9457   \csname glo@#1@firstpluralaccess\endcsname
9458 }

```

`\glsentrysymbolaccess` Get the value of the symbolaccess key for the entry with the given label:

```

9459 \newcommand*\glsentrysymbolaccess[1]{%

```

```
9460 \@gls@entry@field{#1}{symbolaccess}%
9461 }
```

symbolpluralaccess Get the value of the symbolpluralaccess key for the entry with the given label:

```
9462 \newcommand*{\glsentrysymbolpluralaccess}[1]{%
9463 \@gls@entry@field{#1}{symbolpluralaccess}%
9464 }
```

\glsentrydescaccess Get the value of the descriptionaccess key for the entry with the given label:

```
9465 \newcommand*{\glsentrydescaccess}[1]{%
9466 \@gls@entry@field{#1}{descaccess}%
9467 }
```

entrydescpluralaccess Get the value of the descriptionpluralaccess key for the entry with the given label:

```
9468 \newcommand*{\glsentrydescpluralaccess}[1]{%
9469 \@gls@entry@field{#1}{descaccess}%
9470 }
```

\glsentryshortaccess Get the value of the shortaccess key for the entry with the given label:

```
9471 \newcommand*{\glsentryshortaccess}[1]{%
9472 \@gls@entry@field{#1}{shortaccess}%
9473 }
```

entryshortpluralaccess Get the value of the shortpluralaccess key for the entry with the given label:

```
9474 \newcommand*{\glsentryshortpluralaccess}[1]{%
9475 \@gls@entry@field{#1}{shortpluralaccess}%
9476 }
```

\glsentrylongaccess Get the value of the longaccess key for the entry with the given label:

```
9477 \newcommand*{\glsentrylongaccess}[1]{%
9478 \@gls@entry@field{#1}{longaccess}%
9479 }
```

entrylongpluralaccess Get the value of the longpluralaccess key for the entry with the given label:

```
9480 \newcommand*{\glsentrylongpluralaccess}[1]{%
9481 \@gls@entry@field{#1}{longpluralaccess}%
9482 }
```

```
\glsaccsupp \@glsaccsupp{<replacement text>}{<text>}
```

This can be redefined to use E or Alt instead of ActualText. (I don't have the software to test the E or Alt options.)

```
9483 \newcommand*{\glsaccsupp}[2]{%
9484 \BeginAccSupp{ActualText=#1}#2\EndAccSupp{}}%
9485 }
```

`\xglsaccsupp` Fully expands replacement text before calling `\glsaccsupp`

```
9486 \newcommand*\xglsaccsupp}[2]{%
9487   \protected@edef\@gls@replacementtext{#1}%
9488   \expandafter\glsaccsupp\expandafter{\@gls@replacementtext}{#2}%
9489 }
```

`@gls@access@display`

```
9490 \newcommand*\@gls@access@display}[2]{%
9491   \protected@edef\@glo@access{#2}%
9492   \ifx\@glo@access\@gls@noaccess
9493     #1%
9494   \else
9495     \xglsaccsupp{\@glo@access}{#1}%
9496   \fi
9497 }
```

`l$nameaccessdisplay` Displays the first argument with the accessibility text for the entry with the label given by the second argument (if set).

```
9498 \DeclareRobustCommand*\glsnameaccessdisplay}[2]{%
9499   \@gls@access@display{#1}{\glsentryaccess{#2}}%
9500 }
```

`l$textaccessdisplay` As above but for the `textaccess` replacement text.

```
9501 \DeclareRobustCommand*\gls$textaccessdisplay}[2]{%
9502   \@gls@access@display{#1}{\glsentrytextaccess{#2}}%
9503 }
```

`pl$pluralaccessdisplay` As above but for the `pluralaccess` replacement text.

```
9504 \DeclareRobustCommand*\glspluralaccessdisplay}[2]{%
9505   \@gls@access@display{#1}{\glsentrypluralaccess{#2}}%
9506 }
```

`l$firstaccessdisplay` As above but for the `firstaccess` replacement text.

```
9507 \DeclareRobustCommand*\glsfirstaccessdisplay}[2]{%
9508   \@gls@access@display{#1}{\glsentryfirstaccess{#2}}%
9509 }
```

`pl$firstpluralaccessdisplay` As above but for the `firstpluralaccess` replacement text.

```
9510 \DeclareRobustCommand*\glsfirstpluralaccessdisplay}[2]{%
9511   \@gls@access@display{#1}{\glsentryfirstpluralaccess{#2}}%
9512 }
```

`l$symbolaccessdisplay` As above but for the `symbolaccess` replacement text.

```
9513 \DeclareRobustCommand*\gls$symbolaccessdisplay}[2]{%
9514   \@gls@access@display{#1}{\glsentrysymbolaccess{#2}}%
9515 }
```

```

pluralaccessdisplay As above but for the symbolpluralaccess replacement text.
9516 \DeclareRobustCommand*\glssymbolpluralaccessdisplay}[2]{%
9517 \@gls@access@display{#1}{\glstentrysymbolpluralaccess{#2}}}%
9518 }

descriptionaccessdisplay As above but for the descriptionaccess replacement text.
9519 \DeclareRobustCommand*\glsdescriptionaccessdisplay}[2]{%
9520 \@gls@access@display{#1}{\glstentrydescaccess{#2}}}%
9521 }

descriptionpluralaccessdisplay As above but for the descriptionpluralaccess replacement text.
9522 \DeclareRobustCommand*\glsdescriptionpluralaccessdisplay}[2]{%
9523 \@gls@access@display{#1}{\glstentrydescpluralaccess{#2}}}%
9524 }

shortaccessdisplay As above but for the shortaccess replacement text.
9525 \DeclareRobustCommand*\glsshortaccessdisplay}[2]{%
9526 \@gls@access@display{#1}{\glstentryshortaccess{#2}}}%
9527 }

shortpluralaccessdisplay As above but for the shortpluralaccess replacement text.
9528 \DeclareRobustCommand*\glsshortpluralaccessdisplay}[2]{%
9529 \@gls@access@display{#1}{\glstentryshortpluralaccess{#2}}}%
9530 }

longaccessdisplay As above but for the longaccess replacement text.
9531 \DeclareRobustCommand*\glslongaccessdisplay}[2]{%
9532 \@gls@access@display{#1}{\glstentrylongaccess{#2}}}%
9533 }

longpluralaccessdisplay As above but for the longpluralaccess replacement text.
9534 \DeclareRobustCommand*\glslongpluralaccessdisplay}[2]{%
9535 \@gls@access@display{#1}{\glstentrylongpluralaccess{#2}}}%
9536 }

\glsaccessdisplay Gets the replacement text corresponding to the named key given by the first
argument and calls the appropriate command defined above.
9537 \DeclareRobustCommand*\glsaccessdisplay}[3]{%
9538 \@ifundefined{gls#1accessdisplay}%
9539 {%
9540 \PackageError{glossaries-accsupp}{No accessibility support
9541 for key ‘#1’}{}%
9542 }%
9543 {%
9544 \csname gls#1accessdisplay\endcsname{#2}{#3}%
9545 }%
9546 }

```

ls@default@entryfmt Redefine the default entry format to use accessibility information

```
9547 \renewcommand*{\@@gls@default@entryfmt}[2]{%
9548   \ifdefempty\glscustomtext
9549   {%
9550     \glsifplural
9551     {%
```

Plural form

```
9552     \glscapscase
9553     {%
```

Don't adjust case

```
9554     \ifglsused\glslabel
9555     {%
```

Subsequent use

```
9556         #2{\glspluralaccessdisplay
9557           {\glsentryplural{\glslabel}}{\glslabel}}%
9558         {\glsdescriptionpluralaccessdisplay
9559           {\glsentrydescplural{\glslabel}}{\glslabel}}%
9560         {\glsymbolpluralaccessdisplay
9561           {\glsentrysymbolplural{\glslabel}}{\glslabel}}
9562         {\glsinsert}%
9563     }%
9564     {%
```

First use

```
9565         #1{\glsfirstpluralaccessdisplay
9566           {\glsentryfirstplural{\glslabel}}{\glslabel}}%
9567         {\glsdescriptionpluralaccessdisplay
9568           {\glsentrydescplural{\glslabel}}{\glslabel}}%
9569         {\glsymbolpluralaccessdisplay
9570           {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9571         {\glsinsert}%
9572     }%
9573     }%
9574     {%
```

Make first letter upper case

```
9575     \ifglsused\glslabel
9576     {%
```

Subsequent use.

```
9577         #2{\glspluralaccessdisplay
9578           {\Glsentryplural{\glslabel}}{\glslabel}}%
9579         {\glsdescriptionpluralaccessdisplay
9580           {\glsentrydescplural{\glslabel}}{\glslabel}}%
9581         {\glsymbolpluralaccessdisplay
9582           {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9583         {\glsinsert}%
9584     }%
9585     {%
```

First use

```
9586      #1{\glsfirstpluralaccessdisplay
9587          {\Glsentryfirstplural{\glslabel}}{\glslabel}}%
9588          {\glsdescriptionpluralaccessdisplay
9589            {\glsentrydescplural{\glslabel}}{\glslabel}}%
9590          {\glssymbolpluralaccessdisplay
9591            {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9592          {\glsinsert}}%
9593      }%
9594  }%
9595  {%
```

Make all upper case

```
9596      \ifglsused\glslabel
9597      {%
```

Subsequent use

```
9598      \MakeUppercase{%
9599      #2{\glspluralaccessdisplay
9600          {\glsentryplural{\glslabel}}{\glslabel}}%
9601          {\glsdescriptionpluralaccessdisplay
9602            {\glsentrydescplural{\glslabel}}{\glslabel}}%
9603          {\glssymbolpluralaccessdisplay
9604            {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9605          {\glsinsert}}%
9606  }%
9607  {%
```

First use

```
9608      \MakeUppercase{%
9609      #1{\glsfirstpluralaccessdisplay
9610          {\glsentryfirstplural{\glslabel}}{\glslabel}}%
9611          {\glsdescriptionpluralaccessdisplay
9612            {\glsentrydescplural{\glslabel}}{\glslabel}}%
9613          {\glssymbolpluralaccessdisplay
9614            {\glsentrysymbolplural{\glslabel}}{\glslabel}}%
9615          {\glsinsert}}%
9616  }%
9617  }%
9618  }%
9619  {%
```

Singular form

```
9620      \glscapscase
9621      {%
```

Don't adjust case

```
9622      \ifglsused\glslabel
9623      {%
```

Subsequent use

```

9624         #2{\glstextaccessdisplay
9625             {\glentrytext{\glslabel}}{\glslabel}}%
9626         {\glsdescriptionaccessdisplay
9627             {\glentrydesc{\glslabel}}{\glslabel}}%
9628         {\glssymbolaccessdisplay
9629             {\glentrysymbol{\glslabel}}{\glslabel}}%
9630         {\glsinsert}%
9631     }%
9632     {%

```

First use

```

9633         #1{\glsfirstaccessdisplay
9634             {\glentryfirst{\glslabel}}{\glslabel}}%
9635         {\glsdescriptionaccessdisplay
9636             {\glentrydesc{\glslabel}}{\glslabel}}%
9637         {\glssymbolaccessdisplay
9638             {\glentrysymbol{\glslabel}}{\glslabel}}%
9639         {\glsinsert}%
9640     }%
9641 }%
9642 {%

```

Make first letter upper case

```

9643     \ifglsused\glslabel
9644     {%

```

Subsequent use

```

9645         #2{\glstextaccessdisplay
9646             {\glentrytext{\glslabel}}{\glslabel}}%
9647         {\glsdescriptionaccessdisplay
9648             {\glentrydesc{\glslabel}}{\glslabel}}%
9649         {\glssymbolaccessdisplay
9650             {\glentrysymbol{\glslabel}}{\glslabel}}%
9651         {\glsinsert}%
9652     }%
9653     {%

```

First use

```

9654         #1{\glsfirstaccessdisplay
9655             {\glentryfirst{\glslabel}}{\glslabel}}%
9656         {\glsdescriptionaccessdisplay
9657             {\glentrydesc{\glslabel}}{\glslabel}}%
9658         {\glssymbolaccessdisplay
9659             {\glentrysymbol{\glslabel}}{\glslabel}}%
9660         {\glsinsert}%
9661     }%
9662 }%
9663 {%

```

Make all upper case

```

9664     \ifglsused\glslabel
9665     {%

```


Subsequent use

```
9666     \MakeUppercase{%
9667         #2{\glstextaccessdisplay
9668             {\glstentrytext{\glslabel}}{\glslabel}}%
9669         {\glsdescriptionaccessdisplay
9670             {\glstentrydesc{\glslabel}}{\glslabel}}%
9671         {\glssymbolaccessdisplay
9672             {\glstentrysymbol{\glslabel}}{\glslabel}}%
9673         {\glsinsert}}%
9674     }%
9675     {%
```

First use

```
9676     \MakeUppercase{%
9677         #1{\glsfirstaccessdisplay
9678             {\glstentryfirst{\glslabel}}{\glslabel}}%
9679         {\glsdescriptionaccessdisplay
9680             {\glstentrydesc{\glslabel}}{\glslabel}}%
9681         {\glssymbolaccessdisplay
9682             {\glstentrysymbol{\glslabel}}{\glslabel}}%
9683         {\glsinsert}}%
9684     }%
9685     }%
9686     }%
9687     }%
9688     {%
```

Custom text provided in \glsdisp

```
9689     \ifglsused{\glslabel}%
9690     {%
```

Subsequent use

```
9691     #2{\glscustomtext}%
9692     {\glsdescriptionaccessdisplay
9693         {\glstentrydesc{\glslabel}}{\glslabel}}%
9694     {\glssymbolaccessdisplay
9695         {\glstentrysymbol{\glslabel}}{\glslabel}}%
9696     {\glsinsert}%
9697     }%
9698     {%
```

First use

```
9699     #1{\glscustomtext}%
9700     {\glsdescriptionaccessdisplay
9701         {\glstentrydesc{\glslabel}}{\glslabel}}%
9702     {\glssymbolaccessdisplay
9703         {\glstentrysymbol{\glslabel}}{\glslabel}}%
9704     {\glsinsert}%
9705     }%
9706     }%
9707 }
```

`\glsgenentryfmt` Redefine to use accessibility information.

```
9708 \renewcommand*{\glsgenentryfmt}{%
9709   \ifdefempty\glscustomtext
9710   {%
9711     \glsifplural
9712     {%
```

Plural form

```
9713     \glscapscase
9714     {%
```

Don't adjust case

```
9715     \ifglsused\glslabel
9716     {%
```

Subsequent use

```
9717     \glspluralaccessdisplay
9718     {\glsentryplural{\glslabel}}{\glslabel}%
9719     \glsinsert
9720     }%
9721     {%
```

First use

```
9722     \glsfirstpluralaccessdisplay
9723     {\glsentryfirstplural{\glslabel}}{\glslabel}%
9724     \glsinsert
9725     }%
9726     }%
9727     {%
```

Make first letter upper case

```
9728     \ifglsused\glslabel
9729     {%
```

Subsequent use.

```
9730     \glspluralaccessdisplay
9731     {\Glsentryplural{\glslabel}}{\glslabel}%
9732     \glsinsert
9733     }%
9734     {%
```

First use

```
9735     \glsfirstpluralaccessdisplay
9736     {\Glsentryfirstplural{\glslabel}}{\glslabel}%
9737     \glsinsert
9738     }%
9739     }%
9740     {%
```

Make all upper case

```
9741     \ifglsused\glslabel
9742     {%
```

Subsequent use

```
9743      \glspluralaccessdisplay
9744      {\mfirstucMakeUppercase{\glsentryplural{\glslabel}}}%
9745      {\glslabel}%
9746      \mfirstucMakeUppercase{\glsinsert}%
9747      }%
9748      {%
```

First use

```
9749      \glsfirstpluralaccessdisplay
9750      {\mfirstucMakeUppercase{\glsentryfirstplural{\glslabel}}}%
9751      {\glslabel}%
9752      \mfirstucMakeUppercase{\glsinsert}%
9753      }%
9754      }%
9755      }%
9756      {%
```

Singular form

```
9757      \glscapscale
9758      {%
```

Don't adjust case

```
9759      \ifglsused\glslabel
9760      {%
```

Subsequent use

```
9761      \glstextaccessdisplay{\glsentrytext{\glslabel}}{\glslabel}%
9762      \glsinsert
9763      }%
9764      {%
```

First use

```
9765      \glsfirstaccessdisplay{\glsentryfirst{\glslabel}}{\glslabel}%
9766      \glsinsert
9767      }%
9768      }%
9769      {%
```

Make first letter upper case

```
9770      \ifglsused\glslabel
9771      {%
```

Subsequent use

```
9772      \glstextaccessdisplay{\Glsentrytext{\glslabel}}{\glslabel}%
9773      \glsinsert
9774      }%
9775      {%
```

First use

```
9776      \glsfirstaccessdisplay{\Glsentryfirst{\glslabel}}{\glslabel}%
9777      \glsinsert
```

```

9778     }%
9779     }%
9780     {%

```

Make all upper case

```

9781     \ifglsused\glslabel
9782     {%

```

Subsequent use

```

9783     \glstextaccessdisplay
9784     {\mfirstucMakeUppercase{\glsentrytext{\glslabel}}}{\glslabel}%
9785     \mfirstucMakeUppercase{\glsinsert}%
9786     }%
9787     {%

```

First use

```

9788     \glsfirstaccessdisplay
9789     {\mfirstucMakeUppercase{\glsentryfirst{\glslabel}}}{\glslabel}%
9790     \mfirstucMakeUppercase{\glsinsert}%
9791     }%
9792     }%
9793     }%
9794     }%
9795     {%

```

Custom text provided in `\glsdisp`. (The insert should be empty at this point.)
The accessibility information, if required, will have to be explicitly included in
the custom text.

```

9796     \glscustomtext\glsinsert
9797     }%
9798 }

```

`\glsgenacfmt` Redefine to include accessibility information.

```

9799 \renewcommand*{\glsgenacfmt}{%
9800   \ifdefempty\glscustomtext
9801   {%
9802     \ifglsused\glslabel
9803     {%

```

Subsequent use:

```

9804     \glsifplural
9805     {%

```

Subsequent plural form:

```

9806     \glscapscase
9807     {%

```

Subsequent plural form, don't adjust case:

```

9808     \acronymfont
9809     {\glsshortpluralaccessdisplay
9810     {\glsentryshortpl{\glslabel}}{\glslabel}}%
9811     \glsinsert

```

9812 }%
9813 {%

Subsequent plural form, make first letter upper case:

9814 \acronymfont
9815 {\glsshortpluralaccessdisplay
9816 {\Glsentryshortpl{\glslabel}}{\glslabel}}%
9817 \glsinsert
9818 }%
9819 {%

Subsequent plural form, all caps:

9820 \mfirstucMakeUppercase
9821 {\acronymfont
9822 {\glsshortpluralaccessdisplay
9823 {\Glsentryshortpl{\glslabel}}{\glslabel}}%
9824 \glsinsert}%
9825 }%
9826 }%
9827 {%

Subsequent singular form

9828 \glscapscase
9829 {%

Subsequent singular form, don't adjust case:

9830 \acronymfont
9831 {\glsshortaccessdisplay{\Glsentryshort{\glslabel}}{\glslabel}}%
9832 \glsinsert
9833 }%
9834 {%

Subsequent singular form, make first letter upper case:

9835 \acronymfont
9836 {\glsshortaccessdisplay{\Glsentryshort{\glslabel}}{\glslabel}}%
9837 \glsinsert
9838 }%
9839 {%

Subsequent singular form, all caps:

9840 \mfirstucMakeUppercase
9841 {\acronymfont{%
9842 \glsshortaccessdisplay{\Glsentryshort{\glslabel}}{\glslabel}}%
9843 \glsinsert}%
9844 }%
9845 }%
9846 }%
9847 {%

First use:

9848 \glsifplural
9849 {%

First use plural form:

```
9850      \glscapscase
9851      {%
```

First use plural form, don't adjust case:

```
9852      \genplacrfullformat{\glslabel}{\glsinsert}%
9853      }%
9854      {%
```

First use plural form, make first letter upper case:

```
9855      \Genplacrfullformat{\glslabel}{\glsinsert}%
9856      }%
9857      {%
```

First use plural form, all caps:

```
9858      \mfirstucMakeUppercase
9859      {\genplacrfullformat{\glslabel}{\glsinsert}}%
9860      }%
9861      }%
9862      {%
```

First use singular form

```
9863      \glscapscase
9864      {%
```

First use singular form, don't adjust case:

```
9865      \genacrfullformat{\glslabel}{\glsinsert}%
9866      }%
9867      {%
```

First use singular form, make first letter upper case:

```
9868      \Genacrfullformat{\glslabel}{\glsinsert}%
9869      }%
9870      {%
```

First use singular form, all caps:

```
9871      \mfirstucMakeUppercase
9872      {\genacrfullformat{\glslabel}{\glsinsert}}%
9873      }%
9874      }%
9875      }%
9876      }%
9877      {%
```

User supplied text. (The insert should be empty at this point.) The accessibility information, if required, will have to be explicitly included in the custom text.

```
9878      \glscustomtext
9879      }%
9880 }
```

`\genacrfullformat` Redefine to include accessibility information.

```
9881 \renewcommand*{\genacrfullformat}[2]{%
```

```

9882 \glslongaccessdisplay{\glsentrylong{#1}}{#1}#2\space
9883 (\glsshortaccessdisplay{\protect\firstacronymfont{\glsentryshort{#1}}}{#1})%
9884 }

```

`\Genacrfullformat` Redefine to include accessibility information.

```

9885 \renewcommand*{\Genacrfullformat}[2]{%
9886 \glslongaccessdisplay{\Glsentrylong{#1}}{#1}#2\space
9887 (\glsshortaccessdisplay{\protect\firstacronymfont{\Glsentryshort{#1}}}{#1})%
9888 }

```

`\genplacrfullformat` Redefine to include accessibility information.

```

9889 \renewcommand*{\genplacrfullformat}[2]{%
9890 \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}#2\space
9891 (\glsshortpluralaccessdisplay
9892 {\protect\firstacronymfont{\glsentryshortpl{#1}}}{#1})%
9893 }

```

`\Genplacrfullformat` Redefine to include accessibility information.

```

9894 \renewcommand*{\Genplacrfullformat}[2]{%
9895 \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}#2\space
9896 (\glsshortpluralaccessdisplay
9897 {\protect\firstacronymfont{\Glsentryshortpl{#1}}}{#1})%
9898 }

```

`\@acrshort`

```

9899 \def\@acrshort#1#2[#3]{%
9900 \glsdoifexists{#2}%
9901 {%
9902 \let\do@gls@link@checkfirsthyper\relax

9903 \let\glsifplural\@secondoftwo
9904 \let\glsifcaps\@firstofthree
9905 \let\glsinsert\@empty
9906 \def\glscustomtext{%
9907 \acronymfont{\glsshortaccessdisplay{\glsentryshort{#2}}{#2}}#3%
9908 }%

```

Call `\@gls@link`

```

9909 \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9910 }%
9911 }

```

`\@Acrshort`

```

9912 \def\@Acrshort#1#2[#3]{%
9913 \glsdoifexists{#2}%
9914 {%
9915 \let\do@gls@link@checkfirsthyper\relax

```

```

9916 \let\glsifplural\@secondoftwo
9917 \let\glsifcaps\@secondofthree
9918 \let\glsinsert\@empty
9919 \def\glsfont{\acronymfont{\glsfontdisplay{\glsentryshort{#2}}{#2}}#3%
9920 \acronymfont{\glsfontdisplay{\glsentryshort{#2}}{#2}}#3%
9921 }%

```

Call \@gls@link

```

9922 \@gls@link[#1]{#2}{\csname gls@\gls@glstype @entryfmt\endcsname}%
9923 }%
9924 }

```

\@ACRshort

```

9925 \def\@ACRshort#1#2[#3]{%
9926 \glsdoifexists{#2}%
9927 {%
9928 \let\do@gls@link@checkfirsthyper\relax

9929 \let\glsifplural\@secondoftwo
9930 \let\glsifcaps\@thirdofthree
9931 \let\glsinsert\@empty
9932 \def\glsfont{\acronymfont{\glsfontdisplay
9933 \acronymfont{\glsfontdisplay
9934 {\MakeUppercase{\glsentryshort{#2}}}{#2}}#3%
9935 }%

```

Call \@gls@link

```

9936 \@gls@link[#1]{#2}{\csname gls@\gls@glstype @entryfmt\endcsname}%
9937 }%
9938 }

```

\@acrlong

```

9939 \def\@acrlong#1#2[#3]{%
9940 \glsdoifexists{#2}%
9941 {%
9942 \let\do@gls@link@checkfirsthyper\relax

9943 \let\glsifplural\@secondoftwo
9944 \let\glsifcaps\@firstofthree
9945 \let\glsinsert\@empty
9946 \def\glsfont{\acronymfont{\glsfontdisplay{\glsentrylong{#2}}{#2}}#3%
9947 \acronymfont{\glsfontdisplay{\glsentrylong{#2}}{#2}}#3%
9948 }%

```

Call \@gls@link

```

9949 \@gls@link[#1]{#2}{\csname gls@\gls@glstype @entryfmt\endcsname}%
9950 }%
9951 }

```



```

\@AcrLong
9952 \def\@AcrLong#1#2[#3]{%
9953   \glsdoifexists{#2}%
9954   {%
9955     \let\do@gls@link@checkfirsthyper\relax

9956     \let\glsifplural\@secondoftwo
9957     \let\glsifcaps\@firstofthree
9958     \let\glsinsert\@empty
9959     \def\glscustomtext{%
9960       \acronymfont{\glslongaccessdisplay{\Glsentrylong{#2}}{#2}}#3%
9961     }%

    Call \@gls@link
9962     \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9963     }%
9964 }

```

```

\@ACRLong
9965 \def\@ACRLong#1#2[#3]{%
9966   \glsdoifexists{#2}%
9967   {%
9968     \let\do@gls@link@checkfirsthyper\relax

9969     \let\glsifplural\@secondoftwo
9970     \let\glsifcaps\@firstofthree
9971     \let\glsinsert\@empty
9972     \def\glscustomtext{%
9973       \acronymfont{\glslongaccessdisplay{%
9974         \MakeUppercase{\glsentrylong{#2}}}{#2}}#3}%
9975     }%

    Call \@gls@link
9976     \@gls@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
9977     }%
9978 }

```

7.3 Displaying the Glossary

We need to redefine the way the glossary entries are formatted to include the accessibility support. The predefined glossary styles use `\glossentryname`, `\glossentrydesc` and `\glossentrysymbol`, but we need to provide compatibility with earlier versions in case users have defined their own styles using `\accsuppglossaryentryfield` and `\accsuppglossarysubentryfield`.

Now redefine `\glossentryname`, `\glossentrydesc` and `\glossentrysymbol` etc so they use the accessibility stuff.

```

9979 \renewcommand*\glossentryname}[1]{%
9980   \glsdoifexists{#1}%
9981   {%

```

```

9982   \glsnamefont{\glsnameaccessdisplay{\glsentryname{#1}}{#1}}%
9983   }%
9984 }

9985 \renewcommand*{\glossentryname}[1]{%
9986   \glsdoifexists{#1}%
9987   {%
9988     \glsnamefont{\glsnameaccessdisplay{\Glsentryname{#1}}{#1}}%
9989   }%
9990 }

9991 \renewcommand*{\glossentrydesc}[1]{%
9992   \glsdoifexists{#1}%
9993   {%
9994     \glsdescriptionaccessdisplay{\glsentrydesc{#1}}{#1}%
9995   }%
9996 }

9997 \renewcommand*{\Glossentrydesc}[1]{%
9998   \glsdoifexists{#1}%
9999   {%
10000   \glsdescriptionaccessdisplay{\Glsentrydesc{#1}}{#1}%
10001   }%
10002 }

10003 \renewcommand*{\glossentrysymbol}[1]{%
10004   \glsdoifexists{#1}%
10005   {%
10006     \glsymbolaccessdisplay{\glsentrysymbol{#1}}{#1}%
10007   }%
10008 }

10009 \renewcommand*{\Glossentrysymbol}[1]{%
10010   \glsdoifexists{#1}%
10011   {%
10012     \glsymbolaccessdisplay{\Glsentrysymbol{#1}}{#1}%
10013   }%
10014 }

```

pglossaryentryfield

```

10015 \newcommand*{\accsuppglossaryentryfield}[5]{%
10016   \glossaryentryfield{#1}%
10017   {\glsnameaccessdisplay{#2}{#1}}%
10018   {\glsdescriptionaccessdisplay{#3}{#1}}%
10019   {\glsymbolaccessdisplay{#4}{#1}}{#5}%
10020 }

```

glossarysubentryfield

```

10021 \newcommand*{\accsuppglossarysubentryfield}[6]{%
10022   \glossarysubentryfield{#1}{#2}%
10023   {\glsnameaccessdisplay{#3}{#2}}%
10024   {\glsdescriptionaccessdisplay{#4}{#2}}%

```

```
10025 {\glssymbolaccessdisplay{#5}{#2}}{#6}%
10026 }
```

7.4 Acronyms

Redefine acronym styles provided by glossaries:

long-short *<long>* (*<short>*) acronym style.

```
10027 \renewacronymstyle{long-short}%
10028 {%
```

Check for long form in case this is a mixed glossary.

```
10029 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
10030 }%
10031 {%
10032 \renewcommand*{\GenericAcronymFields}{description={\the\glslongtok}}%
10033 \renewcommand*{\genacrfullformat}[2]{%
10034 \glslongaccessdisplay{\glsentrylong{##1}}{##1}##2\space
10035 (\glsshortaccessdisplay
10036 {\protect\firstacronymfont{\glsentryshort{##1}}}{##1})%
10037 }%
10038 \renewcommand*{\Genacrfullformat}[2]{%
10039 \glslongaccessdisplay{\Glsentrylong{##1}}{##1}##2\space
10040 (\glsshortaccessdisplay
10041 {\protect\firstacronymfont{\glsentryshort{##1}}}{##1})%
10042 }%
10043 \renewcommand*{\genplacrfullformat}[2]{%
10044 \glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}##2\space
10045 (\glsshortpluralaccessdisplay
10046 {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1})%
10047 }%
10048 \renewcommand*{\Genplacrfullformat}[2]{%
10049 \glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1}##2\space
10050 (\glsshortpluralaccessdisplay
10051 {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1})%
10052 }%
10053 \renewcommand*{\acronymentry}[1]{%
10054 \glsshortaccessdisplay{acronymfont{\glsentryshort{##1}}}{##1}}
10055 \renewcommand*{\acronymsort}[2]{##1}%
10056 \renewcommand*{\acronymfont}[1]{##1}%
10057 \renewcommand*{\firstacronymfont}[1]{\acronymfont{##1}}%
10058 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10059 }
```

short-long *<short>* (*<long>*) acronym style.

```
10060 \renewacronymstyle{short-long}%
10061 {%
```

Check for long form in case this is a mixed glossary.

```
10062 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
```

```

10063 }%
10064 {%
10065 \renewcommand*\GenericAcronymFields{description={\the\glslongtok}}%
10066 \renewcommand*\genacrfullformat}[2]{%
10067 \glsshortaccessdisplay
10068 {\protect\firstacronymfont{\glsentryshort{##1}}{##1}##2\space
10069 (\glslongaccessdisplay{\glsentrylong{##1}}{##1})%
10070 }%
10071 \renewcommand*\Genacrfullformat}[2]{%
10072 \glsshortaccessdisplay
10073 {\protect\firstacronymfont{\Glsentryshort{##1}}{##1}##2\space
10074 (\glslongaccessdisplay{\glsentrylong{##1}}{##1})%
10075 }%
10076 \renewcommand*\genplacrfullformat}[2]{%
10077 \glsshortpluralaccessdisplay
10078 {\protect\firstacronymfont{\glsentryshortpl{##1}}{##1}##2\space
10079 (\glslongpluralaccessdisplay
10080 {\glsentrylongpl{##1}}{##1})%
10081 }%
10082 \renewcommand*\Genplacrfullformat}[2]{%
10083 \glsshortpluralaccessdisplay
10084 {\protect\firstacronymfont{\Glsentryshortpl{##1}}{##1}##2\space
10085 (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})%
10086 }%
10087 \renewcommand*\acronymentry}[1]{%
10088 \glsshortaccessdisplay{acronymfont{\glsentryshort{##1}}{##1}}%
10089 \renewcommand*\acronymsort}[2]{##1}%
10090 \renewcommand*\acronymfont}[1]{##1}%
10091 \renewcommand*\firstacronymfont}[1]{acronymfont{##1}}%
10092 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%
10093 }

```

long-short-desc *⟨long⟩* (*{⟨short⟩}*) acronym style that has an accompanying description (which the user needs to supply).

```

10094 \renewacronymstyle{long-short-desc}%
10095 {%
10096 \GlsUseAcrEntryDispStyle{long-short}%
10097 }%
10098 {%
10099 \GlsUseAcrStyleDefs{long-short}%
10100 \renewcommand*\GenericAcronymFields{}%
10101 \renewcommand*\acronymsort}[2]{##2}%
10102 \renewcommand*\acronymentry}[1]{%
10103 \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10104 (\glsshortaccessdisplay{acronymfont{\glsentryshort{##1}}{##1}})%
10105 }

```

long-sc-short-desc *⟨long⟩* (\textsc{⟨short⟩}) acronym style that has an accompanying description (which the user needs to supply).

```

10106 \renewacronymstyle{long-sc-short-desc}%
10107 {%
10108   \GlsUseAcrEntryDispStyle{long-sc-short}%
10109 }%
10110 {%
10111   \GlsUseAcrStyleDefs{long-sc-short}%
10112   \renewcommand*\GenericAcronymFields{}%
10113   \renewcommand*\acronymsort}[2]{##2}%
10114   \renewcommand*\acronymentry}[1]{%
10115     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10116     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10117 }

```

long-sm-short-desc *⟨long⟩* (\textsmaller{⟨short⟩}) acronym style that has an accompanying description (which the user needs to supply).

```

10118 \renewacronymstyle{long-sm-short-desc}%
10119 {%
10120   \GlsUseAcrEntryDispStyle{long-sm-short}%
10121 }%
10122 {%
10123   \GlsUseAcrStyleDefs{long-sm-short}%
10124   \renewcommand*\GenericAcronymFields{}%
10125   \renewcommand*\acronymsort}[2]{##2}%
10126   \renewcommand*\acronymentry}[1]{%
10127     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10128     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10129 }

```

short-long-desc *⟨short⟩* ({⟨long⟩}) acronym style that has an accompanying description (which the user needs to supply).

```

10130 \renewacronymstyle{short-long-desc}%
10131 {%
10132   \GlsUseAcrEntryDispStyle{short-long}%
10133 }%
10134 {%
10135   \GlsUseAcrStyleDefs{short-long}%
10136   \renewcommand*\GenericAcronymFields{}%
10137   \renewcommand*\acronymsort}[2]{##2}%
10138   \renewcommand*\acronymentry}[1]{%
10139     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10140     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10141 }

```

sc-short-long-desc *⟨long⟩* (\textsc{⟨short⟩}) acronym style that has an accompanying description (which the user needs to supply).

```

10142 \renewacronymstyle{sc-short-long-desc}%
10143 {%
10144   \GlsUseAcrEntryDispStyle{sc-short-long}%
10145 }%

```

```

10146 {%
10147   \GlsUseAcrStyleDefs{sc-short-long}%
10148   \renewcommand*\GenericAcronymFields{}%
10149   \renewcommand*\acronymsort}[2]{##2}%
10150   \renewcommand*\acronymentry}[1]{%
10151     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10152     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10153 }

```

sm-short-long-desc *<long>* (\textsmaller{<short>}) acronym style that has an accompanying description (which the user needs to supply).

```

10154 \renewacronymstyle{sm-short-long-desc}%
10155 {%
10156   \GlsUseAcrEntryDispStyle{sm-short-long}%
10157 }%
10158 {%
10159   \GlsUseAcrStyleDefs{sm-short-long}%
10160   \renewcommand*\GenericAcronymFields{}%
10161   \renewcommand*\acronymsort}[2]{##2}%
10162   \renewcommand*\acronymentry}[1]{%
10163     \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10164     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10165 }

```

dua *<long>* only acronym style.

```

10166 \renewacronymstyle{dua}%
10167 {%

```

Check for long form in case this is a mixed glossary.

```

10168   \ifdefempty\glscustomtext
10169   {%
10170     \ifglsahaslong{\glslabel}%
10171     {%
10172       \glsifplural
10173       {%

```

Plural form:

```

10174         \glsupcase
10175         {%

```

Plural form, don't adjust case:

```

10176         \glslongpluralaccessdisplay{\glsentrylongpl{\glslabel}}{\glslabel}%
10177         \glsinsert
10178         }%
10179         {%

```

Plural form, make first letter upper case:

```

10180         \glslongpluralaccessdisplay{\Glsentrylongpl{\glslabel}}{\glslabel}%
10181         \glsinsert
10182         }%
10183         {%

```

Plural form, all caps:

```
10184     \glslongpluralaccessdisplay
10185     {\mfirstucMakeUppercase{\glsentrylongpl{\glslabel}}}{\glslabel}%
10186     \mfirstucMakeUppercase{\glsinsert}%
10187     }%
10188     }%
10189     {%
```

Singular form

```
10190     \gls caps case
10191     {%
```

Singular form, don't adjust case:

```
10192     \gls long access display{\glsentrylong{\glslabel}}{\glslabel}\glsinsert
10193     }%
10194     {%
```

Subsequent singular form, make first letter upper case:

```
10195     \gls long access display{\Glsentrylong{\glslabel}}{\glslabel}\glsinsert
10196     }%
10197     {%
```

Subsequent singular form, all caps:

```
10198     \gls long access display
10199     {\mfirstucMakeUppercase
10200     {\glsentrylong{\glslabel}\glsinsert}}{\glslabel}%
10201     \mfirstucMakeUppercase{\glsinsert}%
10202     }%
10203     }%
10204     }%
10205     {%
```

Not an acronym:

```
10206     \gls gen entry fmt
10207     }%
10208     }%
10209     {\gls custom text\glsinsert}%
10210 }%
10211 {%
10212 \renewcommand*{\GenericAcronymFields}{description={\the\gls long tok}}%
10213 \renewcommand*{\acr full fmt}[3]{%
10214     \gls link[##1]{##2}{%
10215         \gls long access display{\glsentrylong{##2}}{##2}##3\space
10216         (\gls short access display{\acronym font{\glsentryshort{##2}}}{##2)}}%
10217 \renewcommand*{\Acr full fmt}[3]{%
10218     \gls link[##1]{##2}{%
10219         \gls long access display{\Glsentrylong{##2}}{##2}##3\space
10220         (\gls short access display{\acronym font{\glsentryshort{##2}}}{##2)}}%
10221 \renewcommand*{\ACR full fmt}[3]{%
10222     \gls link[##1]{##2}{%
10223     \gls long access display
```

```

10224     {\mfirstucMakeUppercase{\glsentrylong{##2}}{##2}##3\space
10225     (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}{##2}})}%
10226 \renewcommand*{\acrfullplfmt}[3]{%
10227   \glslink[##1]{##2}{%
10228     \glslongpluralaccessdisplay
10229     {\glsentrylongpl{##2}}{##2}##3\space
10230     (\glsshortpluralaccessdisplay
10231     {\acronymfont{\glsentryshortpl{##2}}{##2}})}%
10232 \renewcommand*{\Acrfullplfmt}[3]{%
10233   \glslink[##1]{##2}{%
10234     \glslongpluralaccessdisplay
10235     {\Glsentrylongpl{##2}}{##2}##3\space
10236     (\glsshortpluralaccessdisplay
10237     {\acronymfont{\glsentryshortpl{##2}}{##2}})}%
10238 \renewcommand*{\ACRfullplfmt}[3]{%
10239   \glslink[##1]{##2}{%
10240     \glslongpluralaccessdisplay
10241     {\mfirstucMakeUppercase{\glsentrylongpl{##2}}{##2}##3\space
10242     (\glsshortpluralaccessdisplay
10243     {\acronymfont{\glsentryshortpl{##2}}{##2}})}%
10244 \renewcommand*{\glsentryfull}[1]{%
10245   \glslongaccessdisplay{\glsentrylong{##1}}\space
10246   (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}})%
10247 }%
10248 \renewcommand*{\Glsentryfull}[1]{%
10249   \glslongaccessdisplay{\Glsentrylong{##1}}{##1}\space
10250   (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}})%
10251 }%
10252 \renewcommand*{\glsentryfullpl}[1]{%
10253   \glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1}\space
10254   (\glsshortpluralaccessdisplay{\acronymfont{\glsentryshortpl{##1}}{##1}})%
10255 }%
10256 \renewcommand*{\Glsentryfullpl}[1]{%
10257   \glslongpluralaccessdisplay{\Glsentrylongpl{##1}}{##1}\space
10258   (\glsshortpluralaccessdisplay{\acronymfont{\glsentryshortpl{##1}}{##1}})%
10259 }%
10260 \renewcommand*{\acronymentry}[1]{%
10261   \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}}%
10262 \renewcommand*{\acronymsort}[2]{##1}%
10263 \renewcommand*{\acronymfont}[1]{##1}%
10264 \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10265 }

```

dua-desc *<long>* only acronym style with user-supplied description.

```

10266 \renewacronymstyle{dua-desc}%
10267 {%
10268   \GlsUseAcrEntryDispStyle{dua}%
10269 }%
10270 {%

```



```

10271 \GlsUseAcrStyleDefs{dua}%
10272 \renewcommand*\GenericAcronymFields{}%
10273 \renewcommand*\acronymentry}[1]{%
10274     \glslongaccessdisplay{\acronymfont{\glsentrylong{##1}}}{##1}}%
10275 \renewcommand*\acronymsort}[2]{##2}%
10276 }%

```

footnote *<short>*\footnote{*<long>*} acronym style.

```

10277 \renewacronymstyle{footnote}%
10278 {%

```

Check for long form in case this is a mixed glossary.

```

10279 \ifglshaslong{\glslabel}{\glsgenacfmt}{\glsgenentryfmt}%
10280 }%
10281 {%
10282 \renewcommand*\GenericAcronymFields{description={\the\glslongtok}}%

```

Need to ensure hyperlinks are switched off on first use:

```

10283 \glshyperfirstfalse
10284 \renewcommand*\genacrfullformat}[2]{%
10285     \glsshortaccessdisplay
10286     {\protect\firstacronymfont{\glsentryshort{##1}}}{##1}##2%
10287     \protect\footnote{\glslongaccessdisplay{\glsentrylong{##1}}}{##1}}%
10288 }%
10289 \renewcommand*\Genacrfullformat}[2]{%
10290     \glsshortaccessdisplay
10291     {\firstacronymfont{\Glsentryshort{##1}}}{##1}##2%
10292     \protect\footnote{\glslongaccessdisplay{\glsentrylong{##1}}}{##1}}%
10293 }%
10294 \renewcommand*\genplacrfullformat}[2]{%
10295     \glsshortpluralaccessdisplay
10296     {\protect\firstacronymfont{\glsentryshortpl{##1}}}{##1}##2%
10297     \protect\footnote{\glslongpluralaccessdisplay{\glsentrylongpl{##1}}}{##1}}%
10298 }%
10299 \renewcommand*\Genplacrfullformat}[2]{%
10300     \glsshortpluralaccessdisplay
10301     {\protect\firstacronymfont{\Glsentryshortpl{##1}}}{##1}##2%
10302     \protect\footnote{\glslongpluralaccessdisplay{\glsentrylongpl{##1}}}{##1}}%
10303 }%
10304 \renewcommand*\acronymentry}[1]{%
10305     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}%
10306 \renewcommand*\acronymsort}[2]{##1}%
10307 \renewcommand*\acronymfont}[1]{##1}%
10308 \renewcommand*\acrpluralsuffix{\glspluralsuffix}%

```

Don't use footnotes for \acrfull:

```

10309 \renewcommand*\acrfullfmt}[3]{%
10310     \glslink[##1]{##2}{%
10311         \glsshortaccessdisplay{\acronymfont{\glsentryshort{##2}}}{##2}##3\space
10312         (\glslongaccessdisplay{\glsentrylong{##2}}}{##2})}%

```

```

10313 \renewcommand*\Acrfullfmt}[3]{%
10314   \glslink[##1]{##2}{%
10315     \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##2}}{##2}##3\space
10316     (\glslongaccessdisplay{\glsentrylong{##2}}{##2})}%
10317 \renewcommand*\ACRfullfmt}[3]{%
10318   \glslink[##1]{##2}{%
10319     \glsshortaccessdisplay
10320     {\mfirstucMakeUppercase
10321     {\acronymfont{\glsentryshort{##2}}{##2}##3\space
10322     (\glslongaccessdisplay{\glsentrylong{##2}}{##2})}}}%
10323 \renewcommand*\acrfullplfmt}[3]{%
10324   \glslink[##1]{##2}{%
10325     \glsshortpluralaccessdisplay
10326     {\acronymfont{\glsentryshortpl{##2}}{##2}##3\space
10327     (\glslongpluralaccessdisplay{\glsentrylongpl{##2}}{##2})}}}%
10328 \renewcommand*\Acrfullplfmt}[3]{%
10329   \glslink[##1]{##2}{%
10330     \glsshortpluralaccessdisplay
10331     {\acronymfont{\Glsentryshortpl{##2}}{##2}##3\space
10332     (\glslongpluralaccessdisplay{\glsentrylongpl{##2}}{##2})}}}%
10333 \renewcommand*\ACRfullplfmt}[3]{%
10334   \glslink[##1]{##2}{%
10335     \glsshortpluralaccessdisplay
10336     {\mfirstucMakeUppercase
10337     {\acronymfont{\glsentryshortpl{##2}}{##2}##3\space
10338     (\glslongpluralaccessdisplay{\glsentrylongpl{##2}}{##2})}}}%

```

Similarly for \glsentryfull etc:

```

10339 \renewcommand*\glsentryfull}[1]{%
10340   \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}{##1}\space
10341   (\glslongaccessdisplay{\glsentrylong{##1}}{##1})}%
10342 \renewcommand*\Glsentryfull}[1]{%
10343   \glsshortaccessdisplay{\acronymfont{\Glsentryshort{##1}}{##1}\space
10344   (\glslongaccessdisplay{\glsentrylong{##1}}{##1})}%
10345 \renewcommand*\glsentryfullpl}[1]{%
10346   \glsshortpluralaccessdisplay
10347   {\acronymfont{\glsentryshortpl{##1}}{##1}\space
10348   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})}%
10349 \renewcommand*\Glsentryfullpl}[1]{%
10350   \glsshortpluralaccessdisplay
10351   {\acronymfont{\Glsentryshortpl{##1}}{##1}\space
10352   (\glslongpluralaccessdisplay{\glsentrylongpl{##1}}{##1})}%
10353 }

```

footnote-sc \textsc{<short>}\footnote{<long>} acronym style.

```

10354 \renewacronymstyle{footnote-sc}%
10355 {%
10356   \GlsUseAcrEntryDispStyle{footnote}%
10357 }%
10358 {%

```

```

10359 \GlsUseAcrStyleDefs{footnote}%
10360 \renewcommand{\acronymentry}[1]{%
10361     \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
10362 \renewcommand{\acronymfont}[1]{\textsc{##1}}%
10363 \renewcommand*{\acrpluralsuffix}{\glstextup{\glspluralsuffix}}%
10364 }%

```

footnote-sm \textsmaller{<short>}\footnote{<long>} acronym style.

```

10365 \renewacronymstyle{footnote-sm}%
10366 {%
10367     \GlsUseAcrEntryDispStyle{footnote}%
10368 }%
10369 {%
10370     \GlsUseAcrStyleDefs{footnote}%
10371     \renewcommand{\acronymentry}[1]{%
10372         \glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1}}
10373     \renewcommand{\acronymfont}[1]{\textsmaller{##1}}%
10374     \renewcommand*{\acrpluralsuffix}{\glspluralsuffix}%
10375 }%

```

footnote-desc <short>\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

10376 \renewacronymstyle{footnote-desc}%
10377 {%
10378     \GlsUseAcrEntryDispStyle{footnote}%
10379 }%
10380 {%
10381     \GlsUseAcrStyleDefs{footnote}%
10382     \renewcommand*{\GenericAcronymFields}{}%
10383     \renewcommand*{\acronymsort}[2]{##2}%
10384     \renewcommand*{\acronymentry}[1]{%
10385         \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10386         (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10387 }

```

footnote-sc-desc \textsc{<short>}\footnote{<long>} acronym style that has an accompanying description (which the user needs to supply).

```

10388 \renewacronymstyle{footnote-sc-desc}%
10389 {%
10390     \GlsUseAcrEntryDispStyle{footnote-sc}%
10391 }%
10392 {%
10393     \GlsUseAcrStyleDefs{footnote-sc}%
10394     \renewcommand*{\GenericAcronymFields}{}%
10395     \renewcommand*{\acronymsort}[2]{##2}%
10396     \renewcommand*{\acronymentry}[1]{%
10397         \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10398         (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10399 }

```

```

footnote-sm-desc \textsmaller{<short>}\footnote{<long>} acronym style that has an accom-
panying description (which the user needs to supply).
10400 \renewacronymstyle{footnote-sm-desc}%
10401 {%
10402 \GlsUseAcrEntryDispStyle{footnote-sm}%
10403 }%
10404 {%
10405 \GlsUseAcrStyleDefs{footnote-sm}%
10406 \renewcommand*{\GenericAcronymFields}{}%
10407 \renewcommand*{\acronymsort}[2]{##2}%
10408 \renewcommand*{\acronymentry}[1]{%
10409 \glslongaccessdisplay{\glsentrylong{##1}}{##1}\space
10410 (\glsshortaccessdisplay{\acronymfont{\glsentryshort{##1}}}{##1})}%
10411 }

```

Use `\newacronymhook` to modify the key list to set the access text to the long version by default.

```

10412 \renewcommand*{\newacronymhook}{%
10413 \edef\@gls@keylist{shortaccess=\the\glslongtok,%
10414 \the\glskeylisttok}%
10415 \expandafter\glskeylisttok\expandafter{\@gls@keylist}%
10416 }

```

`defaultNewAcronymDef` Modify default style to use access text:

```

10417 \renewcommand*{\DefaultNewAcronymDef}{%
10418 \edef\@do@newglossaryentry{%
10419 \noexpand\newglossaryentry{\the\glslabeltok}%
10420 {%
10421 type=\acronymtype,%
10422 name={\the\glsshorttok},%
10423 description={\the\glslongtok},%
10424 descriptionaccess=\relax,%
10425 text={\the\glsshorttok},%
10426 access={\noexpand\@glo@textaccess},%
10427 sort={\the\glsshorttok},%
10428 short={\the\glsshorttok},%
10429 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10430 shortaccess={\the\glslongtok},%
10431 long={\the\glslongtok},%
10432 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10433 descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10434 first={\noexpand\glslongaccessdisplay
10435 {\the\glslongtok}{\the\glslabeltok}\space
10436 (\noexpand\glsshortaccessdisplay
10437 {\the\glsshorttok}{\the\glslabeltok})},%
10438 plural={\the\glsshorttok\acrpluralsuffix},%
10439 firstplural={\noexpand\glslongpluralaccessdisplay
10440 {\noexpand\@glo@longpl}{\the\glslabeltok}\space
10441 (\noexpand\glsshortpluralaccessdisplay

```

```

10442         {\noexpand\@glo@shortpl}\the\glslabeltok}},%
10443     firstaccess=\relax,
10444     firstpluralaccess=\relax,
10445     textaccess={\noexpand\@glo@shortaccess},%
10446     \the\glskeylisttok
10447 }%
10448 }%
10449 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10450 \let\@org@gls@assign@plural\gls@assign@plural
10451 \let\@org@gls@assign@descplural\gls@assign@descplural
10452 \def\gls@assign@firstpl##1##2{%
10453     \@gls@expand@field{##1}{firstpl}{##2}%
10454 }%
10455 \def\gls@assign@plural##1##2{%
10456     \@gls@expand@field{##1}{plural}{##2}%
10457 }%
10458 \def\gls@assign@descplural##1##2{%
10459     \@gls@expand@field{##1}{descplural}{##2}%
10460 }%
10461 \@do@newglossaryentry
10462 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10463 \let\gls@assign@plural\@org@gls@assign@plural
10464 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10465 }

```

otnoteNewAcronymDef

```

10466 \renewcommand*{\DescriptionFootnoteNewAcronymDef}{%
10467     \edef\@do@newglossaryentry{%
10468         \noexpand\newglossaryentry{\the\glslabeltok}%
10469         {%
10470             type=\acronymtype,%
10471             name={\noexpand\acronymfont{\the\glsshorttok}},%
10472             sort={\the\glsshorttok},%
10473             text={\the\glsshorttok},%
10474             short={\the\glsshorttok},%
10475             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10476             shortaccess={\the\glslongtok},%
10477             long={\the\glslongtok},%
10478             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10479             access={\noexpand\@glo@textaccess},%
10480             plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10481             symbol={\the\glslongtok},%
10482             symbolplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10483             firstpluralaccess=\relax,
10484             textaccess={\noexpand\@glo@shortaccess},%
10485             \the\glskeylisttok
10486         }%
10487     }%
10488     \let\@org@gls@assign@firstpl\gls@assign@firstpl

```

```

10489 \let\@org@gls@assign@plural\gls@assign@plural
10490 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10491 \def\gls@assign@firstpl##1##2{%
10492   \@@gls@expand@field{##1}{firstpl}{##2}%
10493 }%
10494 \def\gls@assign@plural##1##2{%
10495   \@@gls@expand@field{##1}{plural}{##2}%
10496 }%
10497 \def\gls@assign@symbolplural##1##2{%
10498   \@@gls@expand@field{##1}{symbolplural}{##2}%
10499 }%
10500 \@do@newglossaryentry
10501 \let\gls@assign@plural\@org@gls@assign@plural
10502 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10503 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10504 }

```

ptionNewAcronymDef

```

10505 \renewcommand*{\DescriptionNewAcronymDef}{%
10506   \edef\@do@newglossaryentry{%
10507     \noexpand\newglossaryentry{\the\glslabeltok}%
10508     {%
10509       type=\acronymtype,%
10510       name={\noexpand
10511         \acronymformat{\the\glsshorttok}{\the\glslongtok}},%
10512       access={\noexpand\@glo@textaccess},%
10513       sort={\the\glsshorttok},%
10514       short={\the\glsshorttok},%
10515       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10516       shortaccess={\the\glslongtok},%
10517       long={\the\glslongtok},%
10518       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10519       first={\the\glslongtok},%
10520       firstaccess=\relax,
10521       firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10522       text={\the\glsshorttok},%
10523       textaccess={\the\glslongtok},%
10524       plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10525       symbol={\noexpand\@glo@text},%
10526       symbolaccess={\noexpand\@glo@textaccess},%
10527       symbolplural={\noexpand\@glo@plural},%
10528       firstpluralaccess=\relax,
10529       textaccess={\noexpand\@glo@shortaccess},%
10530       \the\glskeylisttok}%
10531   }%
10532 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10533 \let\@org@gls@assign@plural\gls@assign@plural
10534 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10535 \def\gls@assign@firstpl##1##2{%

```

```

10536 \@@gls@expand@field{##1}{firstpl}{##2}%
10537 }%
10538 \def\gls@assign@plural##1##2{%
10539 \@@gls@expand@field{##1}{plural}{##2}%
10540 }%
10541 \def\gls@assign@symbolplural##1##2{%
10542 \@@gls@expand@field{##1}{symbolplural}{##2}%
10543 }%
10544 \@do@newglossaryentry
10545 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10546 \let\gls@assign@plural\@org@gls@assign@plural
10547 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10548 }

```

otnoteNewAcronymDef

```

10549 \renewcommand*{\FootnoteNewAcronymDef}{%
10550 \edef\@do@newglossaryentry{%
10551 \noexpand\newglossaryentry{\the\glslabeltok}%
10552 {%
10553 type=\acronymtype,%
10554 name={\noexpand\acronymfont{\the\glsshorttok}},%
10555 sort={\the\glsshorttok},%
10556 text={\the\glsshorttok},%
10557 textaccess={\the\glslongtok},%
10558 access={\noexpand\@glo@textaccess},%
10559 plural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10560 short={\the\glsshorttok},%
10561 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10562 long={\the\glslongtok},%
10563 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10564 description={\the\glslongtok},%
10565 descriptionplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10566 \the\glskeylisttok
10567 }%
10568 }%
10569 \let\@org@gls@assign@plural\gls@assign@plural
10570 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10571 \let\@org@gls@assign@descplural\gls@assign@descplural
10572 \def\gls@assign@firstpl##1##2{%
10573 \@@gls@expand@field{##1}{firstpl}{##2}%
10574 }%
10575 \def\gls@assign@plural##1##2{%
10576 \@@gls@expand@field{##1}{plural}{##2}%
10577 }%
10578 \def\gls@assign@descplural##1##2{%
10579 \@@gls@expand@field{##1}{descplural}{##2}%
10580 }%
10581 \@do@newglossaryentry
10582 \let\gls@assign@plural\@org@gls@assign@plural

```

```

10583 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10584 \let\gls@assign@descplural\@org@gls@assign@descplural
10585 }

```

\SmallNewAcronymDef

```

10586 \renewcommand*{\SmallNewAcronymDef}{%
10587 \edef\@do@newglossaryentry{%
10588 \noexpand\newglossaryentry{\the\glslabeltok}%
10589 {%
10590 type=\acronymtype,%
10591 name={\noexpand\acronymfont{\the\glsshorttok}},%
10592 access={\noexpand\@glo@symbolaccess},%
10593 sort={\the\glsshorttok},%
10594 short={\the\glsshorttok},%
10595 shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10596 shortaccess={\the\glslongtok},%
10597 long={\the\glslongtok},%
10598 longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10599 text={\noexpand\@glo@short},%
10600 textaccess={\noexpand\@glo@shortaccess},%
10601 plural={\noexpand\@glo@shortpl},%
10602 first={\the\glslongtok},%
10603 firstaccess=\relax,
10604 firstplural={\the\glslongtok\noexpand\acrpluralsuffix},%
10605 description={\noexpand\@glo@first},%
10606 descriptionplural={\noexpand\@glo@firstplural},%
10607 symbol={\the\glsshorttok},%
10608 symbolaccess={\the\glslongtok},%
10609 symbolplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
10610 \the\glskeylisttok
10611 }%
10612 }%
10613 \let\@org@gls@assign@firstpl\gls@assign@firstpl
10614 \let\@org@gls@assign@plural\gls@assign@plural
10615 \let\@org@gls@assign@descplural\gls@assign@descplural
10616 \let\@org@gls@assign@symbolplural\gls@assign@symbolplural
10617 \def\gls@assign@firstpl##1##2{%
10618 \@@gls@expand@field{##1}{firstpl}{##2}%
10619 }%
10620 \def\gls@assign@plural##1##2{%
10621 \@@gls@expand@field{##1}{plural}{##2}%
10622 }%
10623 \def\gls@assign@descplural##1##2{%
10624 \@@gls@expand@field{##1}{descplural}{##2}%
10625 }%
10626 \def\gls@assign@symbolplural##1##2{%
10627 \@@gls@expand@field{##1}{symbolplural}{##2}%
10628 }%
10629 \@do@newglossaryentry

```



```

10630 \let\gls@assign@firstpl\@org@gls@assign@firstpl
10631 \let\gls@assign@plural\@org@gls@assign@plural
10632 \let\gls@assign@descplural\@org@gls@assign@descplural
10633 \let\gls@assign@symbolplural\@org@gls@assign@symbolplural
10634 }

```

The following are kept for compatibility with versions before 3.0:

`\glsshortaccesskey`

```
10635 \newcommand*\glsshortaccesskey{\glsshortkey access}%
```

`\glsshortpluralaccesskey`

```
10636 \newcommand*\glsshortpluralaccesskey{\glsshortpluralkey access}%
```

`\glslongaccesskey`

```
10637 \newcommand*\glslongaccesskey{\glslongkey access}%
```

`\glslongpluralaccesskey`

```
10638 \newcommand*\glslongpluralaccesskey{\glslongpluralkey access}%
```

7.5 Debugging Commands

`\showglongnameaccess`

```

10639 \newcommand*\showglongnameaccess}[1]{%
10640 \expandafter\show\csname glo@\glsdetoklabel{#1}@textaccess\endcsname
10641 }

```

`\showgлотextaccess`

```

10642 \newcommand*\showgлотextaccess}[1]{%
10643 \expandafter\show\csname glo@\glsdetoklabel{#1}@textaccess\endcsname
10644 }

```

`\showglopluralaccess`

```

10645 \newcommand*\showglopluralaccess}[1]{%
10646 \expandafter\show\csname glo@\glsdetoklabel{#1}@pluralaccess\endcsname
10647 }

```

`\showglofirstaccess`

```

10648 \newcommand*\showglofirstaccess}[1]{%
10649 \expandafter\show\csname glo@\glsdetoklabel{#1}@firstaccess\endcsname
10650 }

```

`\showglofirstpluralaccess`

```

10651 \newcommand*\showglofirstpluralaccess}[1]{%
10652 \expandafter\show\csname glo@\glsdetoklabel{#1}@firstpluralaccess\endcsname
10653 }

```

showglosymbolaccess

```
10654 \newcommand*{\showglosymbolaccess}[1]{%
10655   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolaccess\endcsname
10656 }
```

symbolpluralaccess

```
10657 \newcommand*{\showglosymbolpluralaccess}[1]{%
10658   \expandafter\show\csname glo@\glsdetoklabel{#1}@symbolpluralaccess\endcsname
10659 }
```

\showglodescaccess

```
10660 \newcommand*{\showglodescaccess}[1]{%
10661   \expandafter\show\csname glo@\glsdetoklabel{#1}@descaccess\endcsname
10662 }
```

glodescpluralaccess

```
10663 \newcommand*{\showglodescpluralaccess}[1]{%
10664   \expandafter\show\csname glo@\glsdetoklabel{#1}@descpluralaccess\endcsname
10665 }
```

\showgloshortaccess

```
10666 \newcommand*{\showgloshortaccess}[1]{%
10667   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortaccess\endcsname
10668 }
```

loshortpluralaccess

```
10669 \newcommand*{\showgloshortpluralaccess}[1]{%
10670   \expandafter\show\csname glo@\glsdetoklabel{#1}@shortpluralaccess\endcsname
10671 }
```

\showglolongaccess

```
10672 \newcommand*{\showglolongaccess}[1]{%
10673   \expandafter\show\csname glo@\glsdetoklabel{#1}@longaccess\endcsname
10674 }
```

glolongpluralaccess

```
10675 \newcommand*{\showglolongpluralaccess}[1]{%
10676   \expandafter\show\csname glo@\glsdetoklabel{#1}@longpluralaccess\endcsname
10677 }
```

8 Multi-Lingual Support

Many thanks to everyone who contributed to the translations both via email and on comp.text.tex.

8.1 Babel Captions

Define captions if multi-lingual support is required, but the package is not loaded.

```
10678 \NeedsTeXFormat{LaTeX2e}
10679 \ProvidesPackage{glossaries-babel}[2013/11/14 v4.0 (NLCT)]
```

English:

```
10680 \@ifundefined{captionsenglish}{}{%
10681   \addto\captionsenglish{%
10682     \renewcommand*\glossaryname{Glossary}%
10683     \renewcommand*\acronymname{Acronyms}%
10684     \renewcommand*\entryname{Notation}%
10685     \renewcommand*\descriptionname{Description}%
10686     \renewcommand*\symbolname{Symbol}%
10687     \renewcommand*\pagelistname{Page List}%
10688     \renewcommand*\glssymbolsgroupname{Symbols}%
10689     \renewcommand*\glsnumbersgroupname{Numbers}%
10690 }%
10691 }
10692 \@ifundefined{captionsamerican}{}{%
10693   \addto\captionsamerican{%
10694     \renewcommand*\glossaryname{Glossary}%
10695     \renewcommand*\acronymname{Acronyms}%
10696     \renewcommand*\entryname{Notation}%
10697     \renewcommand*\descriptionname{Description}%
10698     \renewcommand*\symbolname{Symbol}%
10699     \renewcommand*\pagelistname{Page List}%
10700     \renewcommand*\glssymbolsgroupname{Symbols}%
10701     \renewcommand*\glsnumbersgroupname{Numbers}%
10702 }%
10703 }
10704 \@ifundefined{captionsaustralian}{}{%
10705   \addto\captionsaustralian{%
10706     \renewcommand*\glossaryname{Glossary}%
10707     \renewcommand*\acronymname{Acronyms}%
10708     \renewcommand*\entryname{Notation}%
10709     \renewcommand*\descriptionname{Description}%
10710     \renewcommand*\symbolname{Symbol}%
10711     \renewcommand*\pagelistname{Page List}%
10712     \renewcommand*\glssymbolsgroupname{Symbols}%
10713     \renewcommand*\glsnumbersgroupname{Numbers}%
10714 }%
10715 }
10716 \@ifundefined{captionsbritish}{}{%
10717   \addto\captionsbritish{%
10718     \renewcommand*\glossaryname{Glossary}%
10719     \renewcommand*\acronymname{Acronyms}%
10720     \renewcommand*\entryname{Notation}%
10721     \renewcommand*\descriptionname{Description}%
```

```

10722 \renewcommand*{\symbolname}{Symbol}%
10723 \renewcommand*{\pagelistname}{Page List}%
10724 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10725 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10726 }}%
10727 \@ifundefined{captionscanadian}{-}{%
10728 \addto\captionscanadian{%
10729 \renewcommand*{\glossaryname}{Glossary}%
10730 \renewcommand*{\acronymname}{Acronyms}%
10731 \renewcommand*{\entryname}{Notation}%
10732 \renewcommand*{\descriptionname}{Description}%
10733 \renewcommand*{\symbolname}{Symbol}%
10734 \renewcommand*{\pagelistname}{Page List}%
10735 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10736 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10737 }}%
10738 }
10739 \@ifundefined{captionsnewzealand}{-}{%
10740 \addto\captionsnewzealand{%
10741 \renewcommand*{\glossaryname}{Glossary}%
10742 \renewcommand*{\acronymname}{Acronyms}%
10743 \renewcommand*{\entryname}{Notation}%
10744 \renewcommand*{\descriptionname}{Description}%
10745 \renewcommand*{\symbolname}{Symbol}%
10746 \renewcommand*{\pagelistname}{Page List}%
10747 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10748 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10749 }}%
10750 }
10751 \@ifundefined{captionsUKenglish}{-}{%
10752 \addto\captionsUKenglish{%
10753 \renewcommand*{\glossaryname}{Glossary}%
10754 \renewcommand*{\acronymname}{Acronyms}%
10755 \renewcommand*{\entryname}{Notation}%
10756 \renewcommand*{\descriptionname}{Description}%
10757 \renewcommand*{\symbolname}{Symbol}%
10758 \renewcommand*{\pagelistname}{Page List}%
10759 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10760 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10761 }}%
10762 }
10763 \@ifundefined{captionsUSenglish}{-}{%
10764 \addto\captionsUSenglish{%
10765 \renewcommand*{\glossaryname}{Glossary}%
10766 \renewcommand*{\acronymname}{Acronyms}%
10767 \renewcommand*{\entryname}{Notation}%
10768 \renewcommand*{\descriptionname}{Description}%
10769 \renewcommand*{\symbolname}{Symbol}%
10770 \renewcommand*{\pagelistname}{Page List}%

```

```

10771 \renewcommand*{\glssymbolsgroupname}{Symbols}%
10772 \renewcommand*{\glsnumbersgroupname}{Numbers}%
10773 }%
10774 }

```

German (quite a few variations were suggested for German; I settled on the following):

```

10775 \@ifundefined{captionsgerman}{}{%
10776 \addto\captionsgerman{%
10777 \renewcommand*{\glossaryname}{Glossar}%
10778 \renewcommand*{\acronymname}{Akronyme}%
10779 \renewcommand*{\entryname}{Bezeichnung}%
10780 \renewcommand*{\descriptionname}{Beschreibung}%
10781 \renewcommand*{\symbolname}{Symbol}%
10782 \renewcommand*{\pagelistname}{Seiten}%
10783 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10784 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10785 }

```

ngerman is identical to German:

```

10786 \@ifundefined{captionsgerman}{}{%
10787 \addto\captionsgerman{%
10788 \renewcommand*{\glossaryname}{Glossar}%
10789 \renewcommand*{\acronymname}{Akronyme}%
10790 \renewcommand*{\entryname}{Bezeichnung}%
10791 \renewcommand*{\descriptionname}{Beschreibung}%
10792 \renewcommand*{\symbolname}{Symbol}%
10793 \renewcommand*{\pagelistname}{Seiten}%
10794 \renewcommand*{\glssymbolsgroupname}{Symbole}%
10795 \renewcommand*{\glsnumbersgroupname}{Zahlen}}
10796 }

```

Italian:

```

10797 \@ifundefined{captionssitalian}{}{%
10798 \addto\captionssitalian{%
10799 \renewcommand*{\glossaryname}{Glossario}%
10800 \renewcommand*{\acronymname}{Acronimi}%
10801 \renewcommand*{\entryname}{Nomenclatura}%
10802 \renewcommand*{\descriptionname}{Descrizione}%
10803 \renewcommand*{\symbolname}{Simbolo}%
10804 \renewcommand*{\pagelistname}{Elenco delle pagine}%
10805 \renewcommand*{\glssymbolsgroupname}{Simboli}%
10806 \renewcommand*{\glsnumbersgroupname}{Numeri}}
10807 }

```

Dutch:

```

10808 \@ifundefined{captionsdutch}{}{%
10809 \addto\captionsdutch{%
10810 \renewcommand*{\glossaryname}{Woordenlijst}%
10811 \renewcommand*{\acronymname}{Acroniemen}%
10812 \renewcommand*{\entryname}{Benaming}%

```

```

10813 \renewcommand*{\descriptionname}{Beschrijving}%
10814 \renewcommand*{\symbolname}{Symbool}%
10815 \renewcommand*{\pagelistname}{Pagina's}%
10816 \renewcommand*{\glssymbolsgroupname}{Symbolen}%
10817 \renewcommand*{\glsnumbersgroupname}{Cijfers}}
10818 }

```

Spanish:

```

10819 \@ifundefined{captionsspanish}{}{%
10820 \addto\captionsspanish{%
10821 \renewcommand*{\glossaryname}{Glosario}%
10822 \renewcommand*{\acronymname}{Siglas}%
10823 \renewcommand*{\entryname}{Entrada}%
10824 \renewcommand*{\descriptionname}{Descripci'on}%
10825 \renewcommand*{\symbolname}{S'\i}mbolo}%
10826 \renewcommand*{\pagelistname}{Lista de p'aginas}%
10827 \renewcommand*{\glssymbolsgroupname}{S'\i}mbolos}%
10828 \renewcommand*{\glsnumbersgroupname}{N'umeros}}
10829 }

```

French:

```

10830 \@ifundefined{captionsfrench}{}{%
10831 \addto\captionsfrench{%
10832 \renewcommand*{\glossaryname}{Glossaire}%
10833 \renewcommand*{\acronymname}{Acronymes}%
10834 \renewcommand*{\entryname}{Terme}%
10835 \renewcommand*{\descriptionname}{Description}%
10836 \renewcommand*{\symbolname}{Symbole}%
10837 \renewcommand*{\pagelistname}{Pages}%
10838 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10839 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10840 }
10841 \@ifundefined{captionsfrenchb}{}{%
10842 \addto\captionsfrenchb{%
10843 \renewcommand*{\glossaryname}{Glossaire}%
10844 \renewcommand*{\acronymname}{Acronymes}%
10845 \renewcommand*{\entryname}{Terme}%
10846 \renewcommand*{\descriptionname}{Description}%
10847 \renewcommand*{\symbolname}{Symbole}%
10848 \renewcommand*{\pagelistname}{Pages}%
10849 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10850 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10851 }
10852 \@ifundefined{captionspancais}{}{%
10853 \addto\captionspancais{%
10854 \renewcommand*{\glossaryname}{Glossaire}%
10855 \renewcommand*{\acronymname}{Acronymes}%
10856 \renewcommand*{\entryname}{Terme}%
10857 \renewcommand*{\descriptionname}{Description}%
10858 \renewcommand*{\symbolname}{Symbole}%

```

```

10859 \renewcommand*{\pagelistname}{Pages}%
10860 \renewcommand*{\glssymbolsgroupname}{Symboles}%
10861 \renewcommand*{\glsnumbersgroupname}{Nombres}}
10862 }

```

Danish:

```

10863 \@ifundefined{captionsdanish}{}{%
10864 \addto\captionsdanish{%
10865 \renewcommand*{\glossaryname}{Ordliste}%
10866 \renewcommand*{\acronymname}{Akronymer}%
10867 \renewcommand*{\entryname}{Symbolforklaring}%
10868 \renewcommand*{\descriptionname}{Beskrivelse}%
10869 \renewcommand*{\symbolname}{Symbol}%
10870 \renewcommand*{\pagelistname}{Side}%
10871 \renewcommand*{\glssymbolsgroupname}{Symboler}%
10872 \renewcommand*{\glsnumbersgroupname}{Tal}}
10873 }

```

Irish:

```

10874 \@ifundefined{captionsirish}{}{%
10875 \addto\captionsirish{%
10876 \renewcommand*{\glossaryname}{Gluais}%
10877 \renewcommand*{\acronymname}{Acrainmneacha}%

```

wasn't sure whether to go for Nóta (Note), Ciall ('Meaning', 'sense') or Brí ('Meaning'). In the end I chose Ciall.

```

10878 \renewcommand*{\entryname}{Ciall}%
10879 \renewcommand*{\descriptionname}{Tuairisc}%

```

Again, not sure whether to use Comhartha/Comharthaí or Siombail/Siombaile, so have chosen the former.

```

10880 \renewcommand*{\symbolname}{Comhartha}%
10881 \renewcommand*{\glssymbolsgroupname}{Comhartha\`{\i}}%
10882 \renewcommand*{\pagelistname}{Leathanaigh}%
10883 \renewcommand*{\glsnumbersgroupname}{Uimhreacha}}
10884 }

```

Hungarian:

```

10885 \@ifundefined{captionsmagyar}{}{%
10886 \addto\captionsmagyar{%
10887 \renewcommand*{\glossaryname}{Sz\`ojegyz\`ek}%
10888 \renewcommand*{\acronymname}{Bet\H uszavak}%
10889 \renewcommand*{\entryname}{Kifejez\`es}%
10890 \renewcommand*{\descriptionname}{Magyar\`azat}%
10891 \renewcommand*{\symbolname}{Jel\`ol\`es}%
10892 \renewcommand*{\pagelistname}{Oldalsz\`am}%
10893 \renewcommand*{\glssymbolsgroupname}{Jelek}%
10894 \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
10895 }
10896 }
10897 \@ifundefined{captionshungarian}{}{%

```

```

10898 \addto\captionshungarian{%
10899   \renewcommand*{\glossaryname}{Sz\`ojegyz\`ek}%
10900   \renewcommand*{\acronymname}{Bet\H uszavak}%
10901   \renewcommand*{\entryname}{Kifejez\`es}%
10902   \renewcommand*{\descriptionname}{Magyar\`azat}%
10903   \renewcommand*{\symbolname}{Jel\`ol\`es}%
10904   \renewcommand*{\pagelistname}{Oldalsz\`am}%
10905   \renewcommand*{\glssymbolsgroupname}{Jelek}%
10906   \renewcommand*{\glsnumbersgroupname}{Sz\`amjegyek}%
10907 }
10908 }

```

Polish

```

10909 \@ifundefined{captionspolish}{}{%
10910   \addto\captionspolish{%
10911     \renewcommand*{\glossaryname}{S{\l}ownik termin\`ow}%
10912     \renewcommand*{\acronymname}{Skr\`ot}%
10913     \renewcommand*{\entryname}{Termin}%
10914     \renewcommand*{\descriptionname}{Opis}%
10915     \renewcommand*{\symbolname}{Symbol}%
10916     \renewcommand*{\pagelistname}{Strony}%
10917     \renewcommand*{\glssymbolsgroupname}{Symbole}%
10918     \renewcommand*{\glsnumbersgroupname}{Liczby}}
10919 }

```

Brazilian

```

10920 \@ifundefined{captionsbrazil}{}{%
10921   \addto\captionsbrazil{%
10922     \renewcommand*{\glossaryname}{Gloss\`ario}%
10923     \renewcommand*{\acronymname}{Siglas}%
10924     \renewcommand*{\entryname}{Nota\c c\~ao}%
10925     \renewcommand*{\descriptionname}{Descri\c c\~ao}%
10926     \renewcommand*{\symbolname}{S\`imbolo}%
10927     \renewcommand*{\pagelistname}{Lista de P\`aginas}%
10928     \renewcommand*{\glssymbolsgroupname}{S\`imbolos}%
10929     \renewcommand*{\glsnumbersgroupname}{N\`umeros}%
10930   }%
10931 }

```

8.2 Polyglossia Captions

```

10932 \NeedsTeXFormat{LaTeX2e}
10933 \ProvidesPackage{glossaries-polyglossia}[2013/11/14 v4.0 (NLCT)]

```

English:

```

10934 \@ifundefined{captionsenglish}{}{%
10935   \expandafter\toks@\expandafter{\captionsenglish
10936     \renewcommand*{\glossaryname}{\textenglish{Glossary}}%
10937     \renewcommand*{\acronymname}{\textenglish{Acronyms}}%
10938     \renewcommand*{\entryname}{\textenglish{Notation}}%
10939     \renewcommand*{\descriptionname}{\textenglish{Description}}%

```



```

10940 \renewcommand*{\symbolname}{\textenglish{Symbol}}%
10941 \renewcommand*{\pagelistname}{\textenglish{Page List}}%
10942 \renewcommand*{\glssymbolsgroupname}{\textenglish{Symbols}}%
10943 \renewcommand*{\glsnumbersgroupname}{\textenglish{Numbers}}%
10944 }%
10945 \edef\captionseenglish{\the\toks@}%
10946 }

German:
10947 \@ifundefined{captionsgerman}{}{%
10948 \expandafter\toks@\expandafter{\captionsgerman
10949 \renewcommand*{\glossaryname}{\textgerman{Glossar}}%
10950 \renewcommand*{\acronymname}{\textgerman{Akronyme}}%
10951 \renewcommand*{\entryname}{\textgerman{Bezeichnung}}%
10952 \renewcommand*{\descriptionname}{\textgerman{Beschreibung}}%
10953 \renewcommand*{\symbolname}{\textgerman{Symbol}}%
10954 \renewcommand*{\pagelistname}{\textgerman{Seiten}}%
10955 \renewcommand*{\glssymbolsgroupname}{\textgerman{Symbole}}%
10956 \renewcommand*{\glsnumbersgroupname}{\textgerman{Zahlen}}%
10957 }%
10958 \edef\captionsgerman{\the\toks@}%
10959 }

Italian:
10960 \@ifundefined{captionssitalian}{}{%
10961 \expandafter\toks@\expandafter{\captionssitalian
10962 \renewcommand*{\glossaryname}{\textitalian{Glossario}}%
10963 \renewcommand*{\acronymname}{\textitalian{Acronimi}}%
10964 \renewcommand*{\entryname}{\textitalian{Nomenclatura}}%
10965 \renewcommand*{\descriptionname}{\textitalian{Descrizione}}%
10966 \renewcommand*{\symbolname}{\textitalian{Simbolo}}%
10967 \renewcommand*{\pagelistname}{\textitalian{Elenco delle pagine}}%
10968 \renewcommand*{\glssymbolsgroupname}{\textitalian{Simboli}}%
10969 \renewcommand*{\glsnumbersgroupname}{\textitalian{Numeri}}%
10970 }%
10971 \edef\captionssitalian{\the\toks@}%
10972 }

Dutch:
10973 \@ifundefined{captionsdutch}{}{%
10974 \expandafter\toks@\expandafter{\captionsdutch
10975 \renewcommand*{\glossaryname}{\textdutch{Woordenlijst}}%
10976 \renewcommand*{\acronymname}{\textdutch{Acroniemen}}%
10977 \renewcommand*{\entryname}{\textdutch{Benaming}}%
10978 \renewcommand*{\descriptionname}{\textdutch{Beschrijving}}%
10979 \renewcommand*{\symbolname}{\textdutch{Symbool}}%
10980 \renewcommand*{\pagelistname}{\textdutch{Pagina's}}%
10981 \renewcommand*{\glssymbolsgroupname}{\textdutch{Symbolen}}%
10982 \renewcommand*{\glsnumbersgroupname}{\textdutch{Cijfers}}%
10983 }%
10984 \edef\captionsdutch{\the\toks@}%

```

10985 }

Spanish:

```
10986 \@ifundefined{captionsspanish}{}{%
10987   \expandafter\toks@\expandafter{\captionsspanish
10988     \renewcommand*{\glossaryname}{\textspanish{Glosario}}%
10989     \renewcommand*{\acronymname}{\textspanish{Siglas}}%
10990     \renewcommand*{\entryname}{\textspanish{Entrada}}%
10991     \renewcommand*{\descriptionname}{\textspanish{Descripci'on}}%
10992     \renewcommand*{\symbolname}{\textspanish{S'\i{mbolo}}}%
10993     \renewcommand*{\pagelistname}{\textspanish{Lista de p'aginas}}%
10994     \renewcommand*{\glssymbolsgroupname}{\textspanish{S'\i{mbolos}}}%
10995     \renewcommand*{\glsnumbersgroupname}{\textspanish{N'umeros}}%
10996   }%
10997   \edef\captionsspanish{\the\toks@}%
10998 }
```

French:

```
10999 \@ifundefined{captionsfrench}{}{%
11000   \expandafter\toks@\expandafter{\captionsfrench
11001     \renewcommand*{\glossaryname}{\textfrench{Glossaire}}%
11002     \renewcommand*{\acronymname}{\textfrench{Acronymes}}%
11003     \renewcommand*{\entryname}{\textfrench{Terme}}%
11004     \renewcommand*{\descriptionname}{\textfrench{Description}}%
11005     \renewcommand*{\symbolname}{\textfrench{Symbole}}%
11006     \renewcommand*{\pagelistname}{\textfrench{Pages}}%
11007     \renewcommand*{\glssymbolsgroupname}{\textfrench{Symboles}}%
11008     \renewcommand*{\glsnumbersgroupname}{\textfrench{Nombres}}%
11009   }%
11010   \edef\captionsfrench{\the\toks@}%
11011 }
```

Danish:

```
11012 \@ifundefined{captionsdanish}{}{%
11013   \expandafter\toks@\expandafter{\captionsdanish
11014     \renewcommand*{\glossaryname}{\textdanish{Ordliste}}%
11015     \renewcommand*{\acronymname}{\textdanish{Akronymer}}%
11016     \renewcommand*{\entryname}{\textdanish{Symbolforklaring}}%
11017     \renewcommand*{\descriptionname}{\textdanish{Beskrivelse}}%
11018     \renewcommand*{\symbolname}{\textdanish{Symbol}}%
11019     \renewcommand*{\pagelistname}{\textdanish{Side}}%
11020     \renewcommand*{\glssymbolsgroupname}{\textdanish{Symboler}}%
11021     \renewcommand*{\glsnumbersgroupname}{\textdanish{Tal}}%
11022   }%
11023   \edef\captionsdanish{\the\toks@}%
11024 }
```

Irish:

```
11025 \@ifundefined{captionsirish}{}{%
11026   \expandafter\toks@\expandafter{\captionsirish
11027     \renewcommand*{\glossaryname}{\textirish{Gluais}}%
11028     \renewcommand*{\acronymname}{\textirish{Acrainmneacha}}%

```

```

11029 \renewcommand*{\entryname}{\textirish{Ciall}}%
11030 \renewcommand*{\descriptionname}{\textirish{Tuaisc}}%
11031 \renewcommand*{\symbolname}{\textirish{Comhartha}}%
11032 \renewcommand*{\glssymbolsgroupname}{\textirish{Comhartha\`{i}}}%
11033 \renewcommand*{\pagelistname}{\textirish{Leathanaigh}}%
11034 \renewcommand*{\glsnumbersgroupname}{\textirish{Uimhreacha}}%
11035 }%
11036 \edef\captionisirish{\the\toks@}%
11037 }

```

Hungarian:

```

11038 \@ifundefined{captionsmagyar}{}{%
11039 \expandafter\toks@\expandafter{\captionsmagyar
11040 \renewcommand*{\glossaryname}{\textmagyar{Sz\`ojegy\`ek}}%
11041 \renewcommand*{\acronymname}{\textmagyar{Bet\H uszavak}}%
11042 \renewcommand*{\entryname}{\textmagyar{Kifejez\`es}}%
11043 \renewcommand*{\descriptionname}{\textmagyar{Magyar\`azat}}%
11044 \renewcommand*{\symbolname}{\textmagyar{Jel\`ol\`es}}%
11045 \renewcommand*{\pagelistname}{\textmagyar{Oldalsz\`am}}%
11046 \renewcommand*{\glssymbolsgroupname}{\textmagyar{Jelek}}%
11047 \renewcommand*{\glsnumbersgroupname}{\textmagyar{Sz\`amjegyek}}%
11048 }%
11049 \edef\captionsmagyar{\the\toks@}%
11050 }

```

Polish

```

11051 \@ifundefined{captionspolish}{}{%
11052 \expandafter\toks@\expandafter{\captionspolish
11053 \renewcommand*{\glossaryname}{\textpolish{S{\l}ownik termin\`ow}}%
11054 \renewcommand*{\acronymname}{\textpolish{Skr\`ot}}%
11055 \renewcommand*{\entryname}{\textpolish{Termin}}%
11056 \renewcommand*{\descriptionname}{\textpolish{Opis}}%
11057 \renewcommand*{\symbolname}{\textpolish{Symbol}}%
11058 \renewcommand*{\pagelistname}{\textpolish{Strony}}%
11059 \renewcommand*{\glssymbolsgroupname}{\textpolish{Symbole}}%
11060 \renewcommand*{\glsnumbersgroupname}{\textpolish{Liczby}}%
11061 }%
11062 \edef\captionspolish{\the\toks@}%
11063 }

```

Portugues

```

11064 \@ifundefined{captionsportuges}{}{%
11065 \expandafter\toks@\expandafter{\captionsportuges
11066 \renewcommand*{\glossaryname}{\textportuges{Gloss\`ario}}%
11067 \renewcommand*{\acronymname}{\textportuges{Siglas}}%
11068 \renewcommand*{\entryname}{\textportuges{Nota\c c\~ao}}%
11069 \renewcommand*{\descriptionname}{\textportuges{Descri\c c\~ao}}%
11070 \renewcommand*{\symbolname}{\textportuges{S\`imbolo}}%
11071 \renewcommand*{\pagelistname}{\textportuges{Lista de P\`aginas}}%
11072 \renewcommand*{\glssymbolsgroupname}{\textportuges{S\`imbolos}}%
11073 \renewcommand*{\glsnumbersgroupname}{\textportuges{N\`umeros}}%

```

```

11074 }%
11075 \edef\captionportuges{\the\toks0}%
11076 }

```

8.3 Brazilian Dictionary

This is a dictionary file provided by Thiago de Melo for use with the package.

```

11077 \ProvidesDictionary{glossaries-dictionary}{Brazilian}

```

Provide Brazilian translations:

```

11078 \providetranslation{Glossary}{Gloss\`ario}
11079 \providetranslation{Acronyms}{Siglas}
11080 \providetranslation{Notation (glossaries)}{Nota\c c\~ao}
11081 \providetranslation{Description (glossaries)}{Descri\c c\~ao}
11082 \providetranslation{Symbol (glossaries)}{S\`imbolo}
11083 \providetranslation{Page List (glossaries)}{Lista de P\`aginas}
11084 \providetranslation{Symbols (glossaries)}{S\`imbolos}
11085 \providetranslation{Numbers (glossaries)}{N\`umeros}

```

8.4 Danish Dictionary

This is a dictionary file provided for use with the package.

```

11086 \ProvidesDictionary{glossaries-dictionary}{Danish}

```

Provide Danish translations:

```

11087 \providetranslation{Glossary}{Ordliste}
11088 \providetranslation{Acronyms}{Akronymer}
11089 \providetranslation{Notation (glossaries)}{Symbolforklaring}
11090 \providetranslation{Description (glossaries)}{Beskrivelse}
11091 \providetranslation{Symbol (glossaries)}{Symbol}
11092 \providetranslation{Page List (glossaries)}{Side}
11093 \providetranslation{Symbols (glossaries)}{Symboler}
11094 \providetranslation{Numbers (glossaries)}{Tal}

```

8.5 Dutch Dictionary

This is a dictionary file provided for use with the package.

```

11095 \ProvidesDictionary{glossaries-dictionary}{Dutch}

```

Provide Dutch translations:

```

11096 \providetranslation{Glossary}{Woordenlijst}
11097 \providetranslation{Acronyms}{Acroniemen}
11098 \providetranslation{Notation (glossaries)}{Benaming}
11099 \providetranslation{Description (glossaries)}{Beschrijving}
11100 \providetranslation{Symbol (glossaries)}{Symbool}
11101 \providetranslation{Page List (glossaries)}{Pagina's}
11102 \providetranslation{Symbols (glossaries)}{Symbolen}
11103 \providetranslation{Numbers (glossaries)}{Cijfers}

```

8.6 English Dictionary

This is a dictionary file provided for use with the package.

```
11104 \ProvidesDictionary{glossaries-dictionary}{English}
```

Provide English translations:

```
11105 \providetranslation{Glossary}{Glossary}
11106 \providetranslation{Acronyms}{Acronyms}
11107 \providetranslation{Notation (glossaries)}{Notation}
11108 \providetranslation{Description (glossaries)}{Description}
11109 \providetranslation{Symbol (glossaries)}{Symbol}
11110 \providetranslation{Page List (glossaries)}{Page List}
11111 \providetranslation{Symbols (glossaries)}{Symbols}
11112 \providetranslation{Numbers (glossaries)}{Numbers}
```

8.7 French Dictionary

This is a dictionary file provided for use with the package.

```
11113 \ProvidesDictionary{glossaries-dictionary}{French}
```

Provide French translations:

```
11114 \providetranslation{Glossary}{Glossaire}
11115 \providetranslation{Acronyms}{Acronymes}
11116 \providetranslation{Notation (glossaries)}{Terme}
11117 \providetranslation{Description (glossaries)}{Description}
11118 \providetranslation{Symbol (glossaries)}{Symbole}
11119 \providetranslation{Page List (glossaries)}{Pages}
11120 \providetranslation{Symbols (glossaries)}{Symboles}
11121 \providetranslation{Numbers (glossaries)}{Nombres}
```

8.8 German Dictionary

This is a dictionary file provided for use with the package.

```
11122 \ProvidesDictionary{glossaries-dictionary}{German}
```

Provide German translations (quite a few variations were suggested for German; I settled on the following):

```
11123 \providetranslation{Glossary}{Glossar}
11124 \providetranslation{Acronyms}{Akronyme}
11125 \providetranslation{Notation (glossaries)}{Bezeichnung}
11126 \providetranslation{Description (glossaries)}{Beschreibung}
11127 \providetranslation{Symbol (glossaries)}{Symbol}
11128 \providetranslation{Page List (glossaries)}{Seiten}
11129 \providetranslation{Symbols (glossaries)}{Symbole}
11130 \providetranslation{Numbers (glossaries)}{Zahlen}
```

8.9 Irish Dictionary

This is a dictionary file provided for use with the package.

```
11131 \ProvidesDictionary{glossaries-dictionary}{Irish}
```

Provide Irish translations:

```
11132 \providetranslation{Glossary}{Gluais}
11133 \providetranslation{Acronyms}{Acrainmneacha}
11134 \providetranslation{Notation (glossaries)}{Ciall}
11135 \providetranslation{Description (glossaries)}{Tuairisc}
11136 \providetranslation{Symbol (glossaries)}{Comhartha}
11137 \providetranslation{Page List (glossaries)}{Leathanaigh}
11138 \providetranslation{Symbols (glossaries)}{Comhartha'\{i}}
11139 \providetranslation{Numbers (glossaries)}{Uimhreacha}
```

8.10 Italian Dictionary

This is a dictionary file provided for use with the package.

```
11140 \ProvidesDictionary{glossaries-dictionary}{Italian}
```

Provide Italian translations:

```
11141 \providetranslation{Glossary}{Glossario}
11142 \providetranslation{Acronyms}{Acronimi}
11143 \providetranslation{Notation (glossaries)}{Nomenclatura}
11144 \providetranslation{Description (glossaries)}{Descrizione}
11145 \providetranslation{Symbol (glossaries)}{Simbolo}
11146 \providetranslation{Page List (glossaries)}{Elenco delle pagine}
11147 \providetranslation{Symbols (glossaries)}{Simboli}
11148 \providetranslation{Numbers (glossaries)}{Numeri}
```

8.11 Magyar Dictionary

This is a dictionary file provided for use with the package.

```
11149 \ProvidesDictionary{glossaries-dictionary}{Magyar}
```

Provide translations:

```
11150 \providetranslation{Glossary}{Sz\'ojegy\ek}
11151 \providetranslation{Acronyms}{Bet\H uszavak}
11152 \providetranslation{Notation (glossaries)}{Kifejez'es}
11153 \providetranslation{Description (glossaries)}{Magyar\'azat}
11154 \providetranslation{Symbol (glossaries)}{Jel\'ol'es}
11155 \providetranslation{Page List (glossaries)}{Oldalsz'am}
11156 \providetranslation{Symbols (glossaries)}{Jelek}
11157 \providetranslation{Numbers (glossaries)}{Sz'amjegyek}
```

8.12 Polish Dictionary

This is a dictionary file provided for use with the package.

```
11158 \ProvidesDictionary{glossaries-dictionary}{Polish}
```

Provide Polish translations:

```
11159 \providetranslation{Glossary}{S{\l}ownik termin\'ow}
11160 \providetranslation{Acronyms}{Skr\'ot}
11161 \providetranslation{Notation (glossaries)}{Termin}
```

```

11162 \providetranslation{Description (glossaries)}{Opis}
11163 \providetranslation{Symbol (glossaries)}{Symbol}
11164 \providetranslation{Page List (glossaries)}{Strony}
11165 \providetranslation{Symbols (glossaries)}{Symbole}
11166 \providetranslation{Numbers (glossaries)}{Liczby}

```

8.13 Serbian Dictionary

This dictionary was provided by Zoran Filipovic.

```

11167 \ProvidesDictionary{glossaries-dictionary}{Serbian}
11168 \providetranslation{Glossary}{Mali re\vnik}
11169 \providetranslation{Acronyms}{Skra\'cenice}
11170 \providetranslation{Notation (glossaries)}{Oznaka}
11171 \providetranslation{Description (glossaries)}{Opis}
11172 \providetranslation{Symbol (glossaries)}{Simbol}
11173 \providetranslation{Page List (glossaries)}{Stranica}
11174 \providetranslation{Symbols (glossaries)}{Simboli}
11175 \providetranslation{Numbers (glossaries)}{Brojevi}

```

8.14 Spanish Dictionary

This is a dictionary file provided for use with the package.

```

11176 \ProvidesDictionary{glossaries-dictionary}{Spanish}

```

Provide Spanish translations:

```

11177 \providetranslation{Glossary}{Glosario}
11178 \providetranslation{Acronyms}{Siglas}
11179 \providetranslation{Notation (glossaries)}{Entrada}
11180 \providetranslation{Description (glossaries)}{Descripci\'on}
11181 \providetranslation{Symbol (glossaries)}{S\'imbolo}
11182 \providetranslation{Page List (glossaries)}{Lista de p\'aginas}
11183 \providetranslation{Symbols (glossaries)}{S\'imbolos}
11184 \providetranslation{Numbers (glossaries)}{N\'umeros}

```

Glossary

`makeindex` An indexing application. [10](#), [24](#), [25](#), [159](#)

`xindy` An flexible indexing application with multilingual support written in Perl. [10](#), [24](#), [25](#), [159](#)

Change History

- ??
 super: fixed typo in `\subglossentry`
 (`\glossentrydesc`) 270
- 1.01
 General: Added range facility in
 format key 97
`\writeist`: Added spaces after
`\delimN` and `\delimR` in ist
 file 144
- 1.03
`\makefirstuc`: changed 'pro-
 tected@edef to 'def 243
- 1.04
 General: Added `\glstextformat` 81
- 1.05
`\glossarysection`: added
`\@mkboth to \glossarysection`
 36
`\gls@defglossaryentry`:
 Changed the default value of
 the sort key to just the value of
 the name key 71
`\glsmakefirstuc`: new 244
- 1.06
 General: now requires etoolbox . 243
`\capitalisewords`: new 245
`\xcapitalisewords`: new 246
- 1.07
`\@gls@link`: fixed bug caused by
`\theglsentrycounter` set-
 ting the page number too soon 94
`\glsadd`: fixed bug caused by
`\theglsentrycounter` set-
 ting the page number too soon
 141
- 1.08
 General: Added babel support ... 31
`\capitalisewords`: made robust
 245
`listgroup`: changed `listgroup`
 style to use `\glsgetgrouptitle`
 252
`altlistgroup`: changed `al-`
`tlistgroup` style to use
`\glsgetgrouptitle` 253
`\makefirstuc`: made robust ... 243
- 1.09
`\@mfu@nocaplist`: new 245
`\capitalisewords`: added check
 for words that shouldn't be
 capitalised 245
`\gMFUnocap`: new 245
`\mfu@checkword`: new 245
`\MFUclear`: new 246
- 1.1
`\@glossarysection`: numbered
 sections and auto label added 38
`\@gls@tmpb`: changed `\toksdef`
 to `\newtoks` 99
`\@gls@toc`: numberline added .. 39
`\@p@glossarysection`: num-
 bered sections and auto label
 added 38
 General: Added support for trans-
 lator package 31
`amsgen` now loaded (`\new@ifnextchar`
 needed) 5
`translate`: `translate` option
 added 22
`\setglossarysection`: new ... 37
`numberedsection`: numbered-
 section package option added . 6
`numberline`: numberline option
 added 6
- 1.12
`\@GLSpl`: now uses `\glsentrydescplural`
 and `\glsentrysymbolplural`
 instead of `\glsentrydesc`
 and `\glsentrysymbol` 111
`\@Glspl@`: now uses `\glsentrydescplural`
 and `\glsentrysymbolplural`
 instead of `\glsentrydesc`
 and `\glsentrysymbol` 110
`\@glspl@`: now uses `\glsentrydescplural`
 and `\glsentrysymbolplural`
 instead of `\glsentrydesc`
 and `\glsentrysymbol` 109
 General: added check for
`\hypertarget` separate to
`\hyperlink` (memoir de-
 fines `\hyperlink` but not
`\hypertarget`) 105

descriptionplural:new	59	\glsnavigation: changed to only use labels for groups that are present	248
\gls@defglossaryentry:					
Changed default first plural to be first key with s appended (was text key with s appended)		71			
descriptionplural support added	71			
symbolplural support added	..	71			
\Glsentrydescplural:New	..	135			
\glsentrydescplural:New	..	135			
\Glsentrysymbolplural:New		136			
\glsentrysymbolplural:New		136			
\SetDescriptionFootnoteAcronymStyle			\gls@hypergroup: new	..	247
Added \protect before \footnote and \glslink		217	\glsnavhypertarget: added check if rerun required	247
\SetFootnoteAcronymStyle:			\glssettoctitle: new	30
Added \protect before \footnote and \glslink		223	\printglossary: changed the way the TOC title is set	166
symbolplural: new	60			
1.13			1.16		
General: Add Polish support	352, 355		\@GLS@: Test glossary type is \acronymtype in addition to checking if footnote option has been used	108
fixed bug that ignored 3rd parameter	112–120	\@GLSpl: Test glossary type is \acronymtype in addition to checking if footnote option has been used	111
\ACRfullpl:new	199	\@Gls@: Test glossary type is \acronymtype in addition to checking if footnote option has been used	107
\Acrfullpl:new	198	\@Glspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used	110
\acrfullpl:new	198	\@gls@: Test glossary type is \acronymtype in addition to checking if footnote option has been used	107
\acrpluralsuffix:New	196	\@glsdisp: Test glossary type is \acronymtype in addition to checking if footnote option has been used	111
\gls@defglossaryentry:			\@glspl@: Test glossary type is \acronymtype in addition to checking if footnote option has been used	109
Changed default first value	..	71	\@glstarget: raised the hypertarget so the target text doesn't		
Changed default firstplural value	71			
Removed restriction on only using \newglossaryentry in the preamble	76			
\newacronym: Removed restriction on only using \newacronym in the preamble		196			
1.14					
\@gls@hypergroup: new	247			
General: added nonnumberlist key to \printglossary	182			
added numberedsection key to \printglossary	180			
\firstacronymfont:new	199			
\glsautoprefix:new	6			
\glsnavhyperlink: changed 'edef to 'protected@edef	...	247			
\glsnavhypertarget: added write to aux file	247			

scroll off the top of the page	105		
\gls@defglossaryentry:			
Changed def to let	71		
1.17			
\@do@wrglossary: new	161		
\@do@seeglossary: new	164		
\@glo@storeentry: new	77		
\@gls@glossary: changed definition to use \index instead of			
\@index	159		
\@glsdefaultplural: new	62		
\@glsdefaultsort: new	63		
\@glsnumber: new	193		
\@glsnoname: new	62		
\@glsnonextpages: new	182		
General: added Brazilian dictionary	356		
Added Brazilian support	352		
added xindy support	24		
parent: new	61		
see: new	60		
\gls@defglossaryentry: added			
nonumberlist key	71		
added parent key	71		
added see key	71		
Stored main part of entry format when entry is defined	76		
\gls@suffiX: new	34		
\gls@suffiXFF: new	35		
\gls@wrglossary: modified to allow for xindy support	159		
\gls@hyperlink: new	141		
\gls@hypernumber: modified to allow material to be attached to location	192		
\glsnavhyperlink: replaced 'hyperlink to '@glslink	247		
\glsnavhypertarget: replaced 'hypertarget to '@glstarget	247		
\glssee: new	165		
\glsseeformat: new	165		
\glsSetSuffiX: new	34		
\glsSetSuffiXFF: new	35		
\ifglxindy: new	24		
\istfilename: added xindy support	33		
\newglossarystyle: made			
\newglossarystyle long	191		
\nopostdesc: new	32		
nonumberlist: new	61		
\printglossary: added check to determine if \printglossary is already defined	166		
added print language to aux file	166		
order: order package option added	24		
\writeist: added xindy support	144		
1.18			
\@gls@loadlist: new	8		
\@gls@loadlong: new	8		
\@gls@loadsuper: new	8		
\@gls@loadtree: new	8		
\gls@defglossaryentry:			
Changed default value of sort to \@glsdefaultsort	71		
moved sort sanitization to \newglossaryentry	75		
\glstarget: new	186		
\oldacronym: new	195		
nolist: new	8		
nolong: new	8		
sort: moved sanitization to \newglossaryentry	59		
nostyles: new	9		
nosuper: new	8		
notree: new	9		
1.19			
\gls@clearpage: new	39		
\gls@disp: new	111		
\SetDescriptionAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller	221		
\SetDescriptionFootnoteAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller	217		
\SetFootnoteAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller	223		
\SetSmallAcronymStyle: changed \acronymfont to use \textsmaller instead of \smaller	226		
1.2			
General: fixed bug in ngerman captions	349		

2.01			
	<code>\@gls@link</code> : moved <code>\@do@wrglossary</code>		
	before term is displayed to prevent unwanted whatsit	95
	<code>\forallglossaries</code> : replaced		
	<code>\ifthenelse</code> with <code>\ifx</code>	48
	<code>\forallglsentries</code> : replaced		
	<code>\ifthenelse</code> with <code>\ifx</code>	49
	<code>\glsdefmain</code> : new	12
	<code>\glsdescwidth</code> : changed		
	<code>\linewidth</code> to <code>\hsize</code>	. 254, 269	
	<code>\glslistdottedwidth</code> : changed		
	<code>\linewidth</code> to <code>\hsize</code>	254
	<code>\glspagelistwidth</code> : changed		
	<code>\linewidth</code> to <code>\hsize</code>	. 254, 269	
	<code>nomain</code> : added <code>nomain</code> package option	13
	<code>\writeist</code> : removed <code>item_02</code> - no such <code>makeindex</code> key	148
2.02			
	<code>\@printglossary</code> : suppressed warning globally rather than locally	169
	General: Changed Brazil to Brazilian	356
	false will prevent automatic loading of translator package		28
	<code>\glossarysection</code> : changed		
	<code>\@mkboth</code> to <code>\glossarymark</code>		36
	<code>\gls glossarymark</code> : New	37
2.03			
	<code>\@GLS@</code> : Added check for hyperfirst	108
	<code>\@GLSp1</code> : Added check for hyperfirst	111
	<code>\@GLs@</code> : Added check for hyperfirst	107
	<code>\@GLspl@</code> : Added check for hyperfirst	110
	<code>\@gls@</code> : Added check for hyperfirst	107
	<code>\@gls@@link</code> : new	94
	<code>\@gls@link</code> : added <code>\leavevmode</code>	95
	Moved entry existence check to avoid duplicate code	95
	<code>\@glsdisp</code> : Added check for hyperfirst	111
	<code>\@glspl@</code> : Added check for hyperfirst	109
	<code>\gls glossarymark</code> : Added check to see if it's already defined	..	37
	<code>hyperfirst</code> : new	22
2.04			
	<code>\@GLS@</code> : Changed test to check if glossary type has been identified as a list of acronyms	...	108
	<code>\@GLSp1</code> : Changed test to check if glossary type has been identified as a list of acronyms	...	111
	<code>\@GLs@</code> : Changed test to check if glossary type has been identified as a list of acronyms	...	107
	<code>\@GLspl@</code> : Changed test to check if glossary type has been identified as a list of acronyms	..	110
	<code>\@glossaryentryfield</code> : new	..	76
	<code>\@glossarysubentryfield</code> : new	77
	<code>\@gls@</code> : Changed test to check if glossary type has been identified as a list of acronyms	...	107
	<code>\@glsacronymlists</code> : new	14
	<code>\@glsdisp</code> : Changed test to check if glossary type has been identified as a list of acronyms	..	111
	<code>\@glspl@</code> : Changed test to check if glossary type has been identified as a list of acronyms	..	109
	<code>\@newglossaryentryposthook</code> : new	76
	<code>\@newglossaryentryprehook</code> : new	76
	<code>acronymlists</code> : new	16
	<code>\DeclareAcronymList</code> : new	..	15
	<code>\DefineAcronymSynonyms</code> : new		211
	<code>\gls@defglossaryentry</code> : added <code>user1-6</code> keys	71
	<code>\glsadd</code> : fixed bug that ignored counter	141
	<code>\Glsentryuseri</code> : new	137
	<code>\glsentryuseri</code> : new	137
	<code>\Glsentryuserii</code> : new	137
	<code>\glsentryuserii</code> : new	137
	<code>\Glsentryuseriii</code> : new	137
	<code>\glsentryuseriii</code> : new	137
	<code>\Glsentryuseriv</code> : new	138

<code>\glstentryuseriv</code> :new	138	3.0	
<code>\Glstentryuseriv</code> :new	138		<code>\@do@wrglossary</code> : added check for hyper location prefix ... 162
<code>\glstentryuseriv</code> :new	138		modified to use new format .. 161
<code>\Glstentryuseriv</code> :new	138		<code>\@glossarysec</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>
<code>\glstentryuseriv</code> :new	138		6
<code>\ns@newglossary</code> : added check to determine if <code>\gls@<type>@display</code> and <code>\gls@<type>@displayfirst</code> have been defined.	56		<code>\do@seeglossary</code> : Sanitize and escape cross-referencing information
<code>\SetAcronymLists</code> :new	15		164
<code>\SetDefaultAcronymDisplayStyle</code> :new	213		<code>\@gls@counterwithin</code> :new ... 10
<code>\SetDefaultAcronymStyle</code> :new	214		<code>\@gls@ifinlist</code> :new
<code>\SetDescriptionAcronymDisplayStyle</code> :new	219		40
<code>\SetDescriptionDUAAcronymDisplayStyle</code> :new	217		<code>\@gls@link</code> : added <code>\@gls@saveentrycounter</code>
<code>\SetDescriptionFootnoteAcronymDisplayStyle</code> :new	215		95
<code>\SetDUADisplayStyle</code> :new ..	227		added <code>\@gls@setsort</code>
<code>\SetFootnoteAcronymDisplayStyle</code> :new	221		95
<code>\SetSmallAcronymDisplayStyle</code> :new	224		<code>\@gls@saveentrycounter</code> :new 96
2.05			<code>\@gls@setupsort@def</code> :new ... 11
<code>\@glsdisp</code> : Added closing brace. Patch provided by Sergiu Dotenco	111		<code>\@gls@setupsort@standard</code> :new
Removed spurious brace. Patch provided by Sergiu Dotenco ..	112		10
<code>\writeist</code> : Added <code>\string</code> before opening and closing braces. Patch provided by Segiu Dotenco	149		<code>\@gls@setupsort@use</code> :new ... 12
2.06			<code>\@gls@locationlist</code> :new 43
<code>\altnewglossary</code> :new	56		<code>\@glslink</code> :replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>
<code>\CustomAcronymFields</code> :new .	229		105
<code>\CustomNewAcronymDef</code> :new .	229		<code>\@glsnextpages</code> :new
<code>\SetCustomDisplayStyle</code> :new	229		183
<code>\SetCustomStyle</code> :new	230		<code>\@makeglossary</code> : Added check for savewrites
2.07			150
General: <code>glsadd</code> format key stored in <code>\@glsnumberformat</code> (was mistakenly stored in <code>\@glo@format</code>)	141		<code>\@print@glossary</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>
			169
			<code>\@printglossary</code> : added <code>\currentglossary</code>
			168
			added <code>\glsnextpages</code>
			168
			make toctitle default to title .. 168
			<code>\@set@glo@numformat</code> : added 4th argument
			97
			<code>\@xdyattributelist</code> :new 40
			General: added prefix to hyperlink
			194
			etoolbox now loaded
			5
			replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>
			29,92,180
			<code>\acrfootnote</code> :new
			215
			<code>\ACRfull</code> : added starred version 198
			<code>\Acrfull</code> : added starred version 197
			<code>\acrfull</code> : added starred version 196
			<code>\ACRfullpl</code> : added starred version
			199
			<code>\Acrfullpl</code> : added starred version
			198

<code>\acrfullpl</code> : added starred version	198	<code>\glshyperlink</code> : changed default from <code>\glstentryname</code> to <code>\glstentrytext</code>	141
<code>\acrlinkfootnote</code> : new	215	<code>\glshypernumber</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	192
<code>\acrnolinkfootnote</code> : new ...	215	<code>\glsnumberformat</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	35
<code>\addglossarytocaptions</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	31	<code>\glsrefentry</code> : new	184
<code>savewrites</code> : new	25	<code>\glsresetsubentrycounter</code> : new	184
<code>see</code> : added <code>\@gls@seeautonumberlist</code>	60	<code>\glsseeitem</code> : hyperlink uses <code>\glsseeitemformat</code> instead of <code>\glstentryname</code>	166
<code>seeautonumberlist</code> : new	8	<code>\glsseeitemformat</code> : new	166
<code>\glossarysection</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	36	<code>\glsortnumberfmt</code> : new	11
<code>\glossarystyle</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	191	<code>\glsstepentry</code> : new	184
<code>\gls@codepage</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	24	<code>\glsstepsubentry</code> : new	184
<code>\gls@defglossaryentry</code> : added <code>\@gls@defsort</code>	75	<code>\glsesubentrycounterlabel</code> : new	185
added short and long keys	71	<code>\glsesubentryitem</code> : new	185
replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	72	<code>theglossary</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	185
<code>\gls@docclearpage</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	38	<code>short</code> : new	62
<code>\gls@wrglossary</code> : modified to take into account <code>savewrites</code>	159	<code>shortplural</code> : new	62
<code>\glsadd</code> : added <code>\@gls@saveentrycounter</code>	142	<code>\ifglossaryexists</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	49
<code>\GlsAddXdyCounters</code> : new	40	<code>\ifglstentryexists</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	50
<code>\glstentrycounterlabel</code> : new	185	<code>\istfile</code> : deprecated	158
<code>\glstentryitem</code> : new	185	<code>glossaryentry</code> : new	183
<code>\Glstentrylong</code> : new	139	<code>glossarysubentry</code> : new	183
<code>\glstentrylong</code> : new	139	<code>\newglossaryentry</code> : replaced <code>\DeclareRobustCommand</code> with <code>\newrobustcmd</code>	65
<code>\Glstentrylongpl</code> : new	139	<code>\newglossarystyle</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	191
<code>\glstentrylongpl</code> : new	139	<code>\ns@newglossary</code> : added <code>\@gls@defsortcount</code>	56
<code>\Glstentryshort</code> : new	138	replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	56
<code>\glstentryshort</code> : new	138	<code>entrycounter</code> : new	9
<code>\Glstentryshortpl</code> : new	139	<code>entrycounterwithin</code> : new	10
<code>\glstentryshortpl</code> : new	138	<code>\oldacronym</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	195
<code>\glsgetgrouptitle</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	189		
<code>\glsglossarymark</code> : replaced <code>\@ifundefined</code> with <code>\ifcsundef</code>	37		

compatible-2.07: compatible-2.07 option added	26	modified to support new format	144
long:new	62	3.01	
longplural:new	62	\@glswritefiles: added check for empty glossaries	158
nonumberlist: now boolean ...	61	General: made robust	108
sort:new	10	\ACRfull: made robust	198
counter: replaced \@ifundefined with \ifcsundef	60	\Acrfull: made robust	197
\printglossary: replaced \@ifundefined with \ifcsundef	166	\acrfull: made robust	196
\SetDescriptionFootnoteAcronymDisplayStyle: expanded options link options	215	\acrfullformat: removed \acronymfont as it should always be set in the second argument	197
\setentrycounter: added optional argument	190	\ACRfullpl: made robust	199
\showacronymlists: new	235	\Acrfullpl: made robust	198
\showglocounter: new	232	\acrfullpl: made robust	198
\showglodesc: new	233	\ACRlong: made robust	130
\showglodescplural: new ...	234	\Acrlong: made robust	129
\showglofirst: new	232	\acrlong: made robust	129
\showglofirstpl: new	232	\ACRlongpl: made robust	132
\showgloflag: new	235	\Acrlongpl: made robust	131
\showgloindex: new	235	\acrlongpl: made robust	131
\showglolevel: new	231	\ACRshort: made robust	127
\showglongame: new	233	\Acrshort: made robust	126
\showgloparent: new	231	\acrshort: made robust	125
\showgloplural: new	231	\ACRshortpl: made robust	128
\showglosort: new	234	\Acrshortpl: made robust	128
\showglossaries: new	235	\acrshortpl: made robust	127
\showglossarycounter: new .	236	\Gls: made robust	107
\showglossaryentries: new .	236	\glsadd: made robust	141
\showglossaryin: new	235	\glsaddall: made robust	142
\showglossaryout: new	236	\GLSdesc: made robust	117
\showglossarytitle: new ...	236	\Glsdesc: made robust	117
\showglosymbol: new	234	\glsdesc: made robust	116
\showglosymbolplural: new .	234	\GLSdescplural: made robust .	118
\showglotext: new	231	\Glsdescplural: made robust .	118
\showglotype: new	232	\glsdescplural: made robust .	117
\showglouseri: new	232	\glsfirst: made robust	113
\showglouserii: new	232	\GLSfirstplural: made robust	115
\showglouseriii: new	233	\Glsfirstplural: made robust	115
\showglouseriv: new	233	\glsfirstplural: made robust	115
\showglouservi: new	233	\glslink: made robust	93
subentrycounter: new	10	\GLSname: made robust	116
\writeist: added xindy-only macro definitions to glossary open tag	146	\Glsname: made robust	116
		\glsname: made robust	116
		\GLSpl: made robust	110
		\Glspl: made robust	109
		\glspl: made robust	109
		\GLSplural: made robust	114

<code>\GLSsymbol</code> : made robust	119	<code>savenumberlist</code> : new	8
<code>\Glsymbol</code> : made robust	119	<code>ucmark</code> : new	9
<code>\glsymbol</code> : made robust	118	<code>\gls@defglossaryentry</code> : added numberlist element	75
<code>\GLSsymbolplural</code> : made robust	120	<code>\gls@save@numberlist</code> : new .	166
<code>\Glsymbolplural</code> : made robust	119	<code>\gls@wrglossary</code> : added check for glossary file defined	160
<code>\glsymbolplural</code> : made robust	119	<code>\glsdisplaynumberlist</code> : new	140
<code>\Glstext</code> : made robust	113	<code>\glsentrycounter</code> : set default value	96
<code>\glstext</code> : made robust	112	<code>\Glsentryfull</code> : fixed bug (re- placed <code>\glsentryshortpl</code> with <code>\glsentryshort</code>)	139
<code>\GLSuseri</code> : made robust	121	<code>\glsentryfullpl</code> : fixed bug (re- placed <code>\glsentryshort</code> with <code>\glsentryshortpl</code>)	139
<code>\Glsuseri</code> : made robust	120	<code>\glsentrynumberlist</code> : new ..	140
<code>\glsuseri</code> : made robust	120	<code>\glsmoveentry</code> : new	76
<code>\GLSuserii</code> : made robust	122	<code>\glsnumlistlastsep</code> : new	141
<code>\Glsuserii</code> : made robust	121	<code>\glsnumlistsep</code> : new	141
<code>\glsuserii</code> : made robust	121	<code>\glsresetsubentrycounter</code> : new	184
<code>\GLSuseriii</code> : made robust	122	<code>\ifglshaschildren</code> : new	51
<code>\Glsuseriii</code> : made robust	122	<code>\ifglshasparent</code> : new	51
<code>\glsuseriii</code> : made robust	122	<code>\makeglossaries</code> : added list parser	153
<code>\GLSuseriv</code> : made robust	123	<code>indexonlyfirst</code> : new	22
<code>\Glsuseriv</code> : made robust	123	<code>\renewglossarystyle</code> : new ..	192
<code>\Glsuseriv</code> : made robust	123	<code>\showglossaryentries</code> : fixed misspelt command	236
<code>\GLSuserv</code> : made robust	124	<code>\SmallNewAcronymDef</code> : fixed broken short and long plural	225
<code>\Glsuserv</code> : made robust	124		
<code>\glsuserv</code> : made robust	124		
<code>\GLSuservi</code> : made robust	125		
<code>\Glsuservi</code> : made robust	125		
<code>\glsuservi</code> : made robust	125		
3.02			
<code>\@do@wrglossary</code> : changed <code>\@glslocref</code> to <code>\theglsentrycounter</code>	163	3.03	
<code>\@do@wrglossary</code> : changed <code>\@do@wr@glossary</code> to test for <code>indexonlyfirst</code> option; put old <code>\@do@wr@glossary</code> code into <code>\@do@wrglossary</code>	160	<code>\@gls@sanitizesort</code> : new	18
<code>\@gls@missingnumberlist</code> : new	62	<code>\@gls@setupsort@standard</code> : used <code>\@gls@sanitizesort</code> .	11
<code>\@gls@writefiles</code> : added check for existence of token in case <code>\makeglossaries</code> has been omitted	158	<code>\@printglossary</code> : allow title to override default <code>toctitle</code>	167
<code>\@printglossary</code> : add a way to fetch current entry label . . .	168	General: allow title to set <code>toctitle</code>	180
General: added check for polyglos- sia	28	<code>\glsinlinedescformat</code> : new .	250
reversed order of package check	32	<code>\glsinlineemptydescformat</code> : new	251
		<code>\glsinlinenameformat</code> : new .	250
		<code>\glsinlinepostchild</code> : new ..	250
		<code>\glsinlinesubdescformat</code> : new	251
		<code>\glsinlinesubnameformat</code> : new	251

<code>\glspostinline</code> : replaced “.” with <code>\glspostdescription</code>	250		
<code>altragged4col</code> : added check for <code>glsnogroupskip</code>	.. 264		
<code>altsuperragged4col</code> : added check for <code>glsnogroupskip</code>	.. 281		
<code>alttree</code> : added check for <code>glsnogroupskip</code> 289		
<code>index</code> : added check for <code>glsnogroupskip</code> 283		
<code>nogroupskip</code> : new 9		
<code>long</code> : added check for <code>glsnogroupskip</code> 255		
<code>long3col</code> : added check for <code>glsnogroupskip</code> 256		
<code>long4col</code> : added check for <code>glsnogroupskip</code> 258		
<code>longragged</code> : added check for <code>glsnogroupskip</code> 261		
<code>longragged3col</code> : added check for <code>glsnogroupskip</code> 262		
<code>nopostdot</code> : new 9		
<code>tree</code> : added check for <code>glsnogroupskip</code> 284		
<code>treename</code> : added check for <code>glsnogroupskip</code> 286		
<code>super</code> : added check for <code>glsnogroupskip</code> 270		
<code>super3col</code> : added check for <code>glsnogroupskip</code> 272		
<code>super4col</code> : added check for <code>glsnogroupskip</code> 273		
<code>superragged</code> : added check for <code>glsnogroupskip</code> 277		
<code>superragged3col</code> : added check for <code>glsnogroupskip</code> 279		
3.04			
<code>\@do@wrglossary</code> : changed <code>\theglscopycounter</code> back to <code>\@glslocref</code> 163		
<code>\@do@wrglossary</code> : modified to compensate for possible incor- rect page number 161		
<code>\@gls@escbsdq</code> : unsani- tize <code>\gls@numberpage</code> , <code>\gls@alphpage</code> , <code>\gls@Alphpage</code> and <code>\gls@romanpage</code> 98		
<code>\@print@glossary</code> : Moved aux write to end of document to prevent unwanted whatsit oc- curring here. 169		
General: Added check for doc package 5		
added datatool-base as a re- quired package 5		
added local key 92		
<code>\gls@Alphpage</code> : new 160		
<code>\gls@alphpage</code> : new 160		
<code>\gls@disablepagerefexpansion</code> : new 160		
<code>\gls@numberpage</code> : new 161		
<code>\gls@protected@pagefmts</code> : new 160		
<code>\gls@romanpage</code> : new 161		
<code>\glsdefmain</code> : added check for doc package 12		
<code>\glsorg@endtheglossary</code> : new	.. 5		
<code>\glsorg@theglossary</code> : new 5		
<code>altlist</code> : replaced <code>\newline</code> with paragraph break 252		
<code>\PrintChanges</code> : new 5		
3.05			
<code>\@do@wrglossary</code> : add Roman case. Fixed bugs in the else statements 162		
<code>\@gls@link</code> : added check for “no- hypertypes” 95		
<code>\@gls@nohyperlist</code> : new 16		
<code>mcolalttree</code> : replaced ‘2’ with <code>\glscols</code> 268		
<code>mcolindex</code> : replaced ‘2’ with <code>\glscols</code> 266		
<code>mcoltree</code> : replaced ‘2’ with <code>\glscols</code> 266		
<code>mcoltreename</code> : replaced ‘2’ with <code>\glscols</code> 267		
<code>\gls@protected@pagefmts</code> : added Roman to list 160		
<code>\gls@Romanpage</code> : new 161		
<code>\GlsDeclareNoHyperList</code> : new	16		
<code>\glsgetgrouplabel</code> : fixed bug (typo in <code>\equal</code>) 190		
<code>\nopostdesc</code> : made robust 32		
<code>nohypertypes</code> : new 16		
3.06			
<code>\@xdy@main@language</code> : Changed back to using <code>\language</code>	24		
<code>\findrootlanguage</code> : Obsoleted	47		

3.07		
	<code>\@gls@link</code> : fixed bug that failed to find entry in list	95
	<code>\glossarypreamble</code> : modified to work with <code>\setglossarypreamble</code>	36
	<code>\gls@doclearpage</code> : added check for <code>openright</code>	38
	<code>\glspostdescription</code> : Added <code>spacefactor</code> code	9
	<code>\GlsSetXdyCodePage</code> : Added check for <code>fontspec</code>	48
	<code>\SetDescriptionAcronymDisplayStyle</code> : now using <code>\glsdoparenifnotempty</code>	219
	<code>\setglossarypreamble</code> : new	36
3.08a		
	<code>\@glo@storeentry</code> : no longer need to check for special characters in any of the fields other than <code>sort</code>	77
	updated for <code>\glossentry</code>	77
	<code>\@glossaryentryfield</code> : switched to <code>\glossentry</code>	76
	<code>\@glossarysubentryfield</code> : switched to <code>\subglossentry</code>	77
	General: added <code>nogroupskip</code> key to <code>\printglossary</code>	180
	removed definition of <code>\@glossaryentryfield</code>	329
	removed definition of <code>\@glossarysubentryfield</code>	329
	<code>\compatibleglossentry</code> : new	186
	<code>\compatiblesubglossentry</code> : new	187
	<code>\glossaryentryfield</code> : deprecated	188
	<code>\Glossentrydesc</code> : new	187
	<code>\glossentrydesc</code> : new	187
	<code>\Glossentryname</code> : new	186
	<code>\glossentryname</code> : new	186
	<code>\Glossentrysymbol</code> : new	187
	<code>\glossentrysymbol</code> : new	187
	<code>\gls@assign@desc@field</code> : new	18
	<code>\gls@assign@descplural@field</code> : new	18
	<code>\gls@assign@field</code> : new	64
	<code>\gls@ifnotmeasuring</code> : new	78
	<code>\glsaddallunused</code> : new	142
	<code>\glsexpandfields</code> : new	65
	<code>\glsnoexpandfields</code> : new	65
	<code>\glssee</code> : made robust	165
	<code>\glsseeformat</code> : made robust	165
	<code>\glsseeitem</code> : made robust	166
	<code>\glsseelist</code> : made robust	165
	<code>\ifglsdescsuppressed</code> : new	52
	<code>\ifglshasdesc</code> : new	52
	<code>\ifglshassymbol</code> : new	52
	<code>list</code> : updated list style to use <code>\glossentry</code> and <code>\subglossentry</code>	251
	<code>listdotted</code> : updated listdotted style to use <code>\glossentry</code> and <code>\subglossentry</code>	253
	<code>altlist</code> : updated altlist style to use <code>\glossentry</code> and <code>\subglossentry</code>	252
	<code>altlongragged4col</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code>	264
	<code>alttree</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code>	287
	<code>index</code> : added paragraph break at end of environment	282
	updated to use <code>\glossentry</code> and <code>\subglossentry</code>	282
	<code>inline</code> : updated inline style to use <code>\glossentry</code> and <code>\subglossentry</code>	249
	<code>long</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code>	255
	<code>longragged</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code>	261
	<code>longragged3col</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code>	262
	<code>tree</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code>	284
	<code>\setglossarystyle</code> : new	191
	<code>\setglossentrycompatibility</code> : new	188
	<code>superragged</code> : updated to use <code>\glossentry</code> and <code>\subglossentry</code>	277

3.09a			
	\@gls@assign@symbolplural@field:		\gls.writedefhook:new 70
	new	18	\makeglossaries: Added
	\@gls@default@value:new ...	59	providecommand code to aux
	\Glsentrydesc:made robust ..	135	file
	\Glsentrydescplural:made ro-		152
	bust	135	\new@glossaryentry:new 65
	\Glsentryfirst:made robust .	136	\ns@newglossary: added
	\Glsentryfirstplural: made		\@gls@provide@newglossary
	robust	136 56
	\Glsentryfull:made robust ..	139	3.11a
	\Glsentryfullpl:made robust	139	\@ACRlong: added \glslabel,
	\Glsentrylong:made robust ..	139	\glsifplural,\glscapscase,
	\Glsentrylongpl:made robust	139	\glsinsert and \glscustomtext
	\Glsentryname:made robust ..	134 329
	\Glsentryplural:made robust	135	\@ACRshort: added \glslabel,
	\Glsentryshort:made robust .	138	\glsifplural,\glscapscase,
	\Glsentryshortpl:made robust		\glsinsert and \glscustomtext
	139 328
	\Glsentrysymbol:made robust	136	\@Acrlong: added \glslabel,
	\Glsentrysymbolplural: made		\glsifplural,\glscapscase,
	robust	136	\glsinsert and \glscustomtext
	\Glsentrytext:made robust ..	135 329
	\Glsentryuseri:made robust .	137	\@Acrshort: added \glslabel,
	\Glsentryuserii:made robust	137	\glsifplural,\glscapscase,
	\Glsentryuseriii:made robust		\glsinsert and \glscustomtext
	137 328
	\Glsentryuseriv:made robust	138	\@GLS@: add \glslabel,
	\Glsentryuserv:made robust .	138	\glsifplural,\glscapscase,
	\Glsentryuservi:made robust	138	\glscustomtext and
	\glstextup:new	196	\glsinsert
	\ifglshassymbol: changed test		108
	to check for \@gls@default@symbol		change to using \glsentryfmt
	52	style commands
			108
			removed \MakeUppercase
			(now moved to \glsentryfmt)
		 108
3.10a			\@GLSpl: add \glslabel,
	\@gls@keymap:new	66	\glsifplural,\glscapscase,
	\@gls@provide@newglossary:		\glscustomtext and
	new	54	\glsinsert
	\@gls@writedef:new	66 111
	\@gls@defaultplural:Obsolete .	62	change to using \glsentryfmt
	\@gls@nodesc:new	62	style commands
	\@print@glossary: Added		111
	providecommand code to aux		removed \MakeUppercase
	file	169, 170	as now dealt with in
	\gls@assign@type@field:new	18	\glsentryfmt
	\gls@defglossaryentry:		111
	Changed to using \@gls@default@value		\@Gls@: add \glsifplural,
	71	\glscapscase,\glscustomtext
	new	71	and \glsinsert
		 107
			change to using \glsentryfmt
			style commands
			107

removed \makefirstuc (now dealt with in \glentryfmt) 107	changed to just use \glentrydesc 117
\@Glspl@: add \glsifplural, \glscapscase, \glscustomtext and \glsinsert 110	changed to just use \Glsentryfirstplural 115
change to using \glentryfmt style commands 110	changed to just use \glentryfirstplural 115
removed \makefirstuc (now dealt with in \glentryfmt) 110	changed to just use \Glsentryfirst 113
\@acrlong: added \glslabel, \glsifplural, \glscapscase, \glsinsert and \glscustomtext 328	changed to just use \glentryfirst 113, 114
\@acrshort: added \glslabel, \glsifplural, \glscapscase, \glsinsert and \glscustomtext 327	changed to just use \Glsentryname 116
\@gls@: add \glslabel, \glsifplural, \glscapscase, \glscustomtext and \glsinsert 106	changed to just use \glentryname 116
change to using \glentryfmt style commands 107	changed to just use \Glsentryplural 114
\@gls@noexpand@fields: Fixed bug expand replaced with noexpand 63	changed to just use \glentryplural 114, 115
\@glsdisp: add \glslabel, \glsifplural, \glscapscase, \glscustomtext and \glsinsert 111	changed to just use \Glsentrysymbolplural 120
change to using \glentryfmt style commands 111	changed to just use \glentrysymbolplural 119, 120
\@Glspl@: add \glslabel, \glsifplural, \glscapscase, \glscustomtext and \glsinsert 109	changed to just use \Glsentrysymbol 119
change to using \glentryfmt style commands 109	changed to just use \glentrysymbol 118, 119
General: added \glslabel, \glsifplural, \glscapscase, \glsinsert and \glscustomtext 126–132	Changed to just use \Glsentrytext 113
changed to just use \Glsentrydescplural 118	changed to just use \glentrytext 112
changed to just use \glentrydescplural 118	changed to just use \Glsentryuseriii 122
changed to just use \Glsentrydesc 117	changed to just use \glentryuseriii 122, 123
	changed to just use \Glsentryuserii 121
	changed to just use \glentryuserii 121, 122
	changed to just use \Glsentryuseriv 123
	changed to just use \glentryuseriv 123, 124
	changed to just use \Glsentryuseri 121
	changed to just use \glentryuseri 120, 121
	changed to just use \Glsentryuseriv 125

changed to just use <code>\glsentryuservi</code>	229
.....	125
changed to just use <code>\Glsentryuserv</code>	124
.....	124
changed to just use <code>\glsentryuserv</code>	124
.....	124
Now requires textcase	4
acronymlists: replaced	
<code>\@addtoacronymlists</code> with	
<code>\DeclareAcronymList</code>	16
<code>\defglsdisplay</code> : obsoleted	90
<code>\defglsdisplayfirst</code> : obso-	
leted	91
<code>\defglsentryfmt</code> : new	54
<code>\for glossaries</code> : replaced <code>\ifx</code>	
with <code>\ifdefempty</code>	49
<code>\gls@assign@desc</code> : new	70
<code>\gls@defglossaryentry</code> : Fixed	
default counter if none sup-	
plied	74
<code>\gls@doentryfmt</code> : new	54
<code>\glsdisplay</code> : obsoleted	90
<code>\glsdisplayfirst</code> : obsoleted	90
<code>\glsgenentryfmt</code> : new	85
<code>\glsgetgrouptitle</code> : Added	
check in case non-Latin alpha-	
bet in use	189
<code>\gls glossarymark</code> : replaced	
<code>\MakeUppercase</code> with	
<code>\mfirstucMakeUppercase</code>	37
<code>\glsnavigation</code> : switched to us-	
ing <code>\@gls@getgrouptitle</code>	248
<code>\ifgls has desc</code> : replaced	
<code>\ifdefempty</code> with <code>\ifcsemt</code>	
pty	52
<code>\ifgls has long</code> : new	52
<code>\ifgls has short</code> : new	52
<code>\ifgls has symbol</code> : replaced	
<code>\ifdefempty</code> with <code>\ifcsemt</code>	
pty	52
<code>\ifgls used</code> : replaced <code>\ifthenelse</code>	
with <code>\ifbool</code>	50
<code>\long new glossaryentry</code> : new	70
<code>\ns@new glossary</code> : replaced	
<code>\glsdisplay</code> and <code>\glsdisplayfirst</code>	
with <code>\glsentryfmt</code>	56
compatible-3.07: cnew	26
<code>\SetCustomDisplayStyle</code> : up-	
dated to use <code>\defglsentryfmt</code>	
<code>\SetDefaultAcronymDisplayStyle</code> :	
changed to use <code>\defglsentryfmt</code>	213
<code>\SetDescriptionAcronymDisplayStyle</code> :	
updated to use <code>\defglsentryfmt</code>	219
<code>\SetDescriptionDUAAcronymDisplayStyle</code> :	
updated to use <code>\defglsentryfmt</code>	217
<code>\SetDescriptionFootnoteAcronymDisplayStyle</code> :	
updated to use <code>\defglsentryfmt</code>	215
<code>\SetDUADisplayStyle</code> : updated	
to use <code>\defglsentryfmt</code>	227
<code>\SetFootnoteAcronymDisplayStyle</code> :	
updated to use <code>\defglsentryfmt</code>	221
<code>\SetSmallAcronymDisplayStyle</code> :	
updated to use <code>\defglsentryfmt</code>	224
<code>\setupglossaries</code> : new	27
<code>\showglo long</code> : new	234
<code>\showglo short</code> : new	234
numbers: new	26
symbols: new	26
3.12a	
<code>\gls@defglossaryentry</code> : added	
<code>\glslabel</code>	71
<code>\glsaddkey</code> : new	68
3.13a	
<code>\@gls@assign@symbol@field</code> :	
changed to use <code>\glssetnoexpandfield</code>	18
<code>\@gls@assign@symbolplural@field</code> :	
changed to use <code>\glssetnoexpandfield</code>	18
<code>\@gls@link</code> : removed <code>\relax</code>	96
<code>\@gls@notranslatorhook</code> : new	22
<code>\@gls@setupsort@standard</code> :	
moved <code>\@gls@santizesort</code>	
to <code>\glsprestandardsort</code>	11
General: added <code>cs@gls@notranslatorhook</code>	
to else clause	32
ucmark: added check for memoir	9
see: added <code>\gls@checkseeallowed</code>	60
<code>\glossarysection</code> : changed	
<code>\glossarymark</code> to <code>\gls glossarymark</code>	

..... 36	long4colheader: switched to
\glossarystyle: fixed bug	\tabularnewline 258
caused by using \ifdef in-	longheader: switched to
stead of \ifcsdef 191	\tabularnewline 255
\gls@assign@desc@field:	longheaderborder: switched to
changed to use \glssetnoexpandfield	\tabularnewline 256
..... 18	\SetFootnoteAcronymDisplayStyle:
\gls@assign@descplural@field:	fixed missing argument bug 222
changed to use \glssetnoexpandfieldsuper:	switched to \tabularnewline
..... 18 270
\gls@assign@name@field:	super3col: switched to
changed to use \glssetnoexpandfield	\tabularnewline 271
..... 18	super3colheader: switched to
\gls@assign@type@field:	\tabularnewline 272
changed to use \glssetexpandfield	super4col: switched to
..... 18	\tabularnewline 273
\gls@checkseeallowed: new .. 60	super4colheader: switched to
\glsaddallunused: set default to	\tabularnewline 274
\@glo@types 142	super4colheaderborder:
\Glsentryfull: changed to use	switched to \tabularnewline
\acrfullformat 139 274
\glsentryfull: changed to use	superheader: switched to
\acrfullformat 139	\tabularnewline 271
\Glsentryfullpl: changed to	superheaderborder: switched to
use \acrfullformat 139	\tabularnewline 271
\glsentryfullpl: changed to	3.14a
use \acrfullformat 139	\@glswritefiles: renamed
\gls glossarymark: renamed	\glswritefiles to \@glswritefiles
\glossarymark to \gls glossarymark	and used "savewrites" option
to avoid conflict with memoir 37	to set \glswritefiles 158
\glsprestandardsort: new ... 10	General: new 237
\glssetexpandfield: new 17	acronyms: new 14
\glssetnoexpandfield: new .. 17	\gls@def glossaryentry: added
altsuper4colheader: switched	check for existence of default
to \tabularnewline 275	glossary 72
altsuper4colheaderborder:	set the default for firstplural to
switched to \tabularnewline	be the value of plural 74
..... 276	xindygloss: new 25
long: switched to \tabularnewline	\longprovideglossaryentry:
..... 255	new 70
long3col: switched to \tabularnewline	compatible-2.07: added check
..... 256	for 2.07 before setting 3.07
long3colheader: switched to	compatibility 26
\tabularnewline 257	notranslate: new 22
long3colheaderborder: switched	\provideglossaryentry: new . 65
to \tabularnewline 257	4.0
long4col: switched to \tabularnewline	\gls@def glossaryentry: added
..... 257	check for first key 74

4.01			
	General: fixed non-value options so that they can be passed to document class	7	
	\CustomAcronymFields: in- serted missing comma	229	
4.02			
	\@acrfull: now using \acrfullfmt	197	
	\@gls@indexdef: new	27	
	\@gls@numbersdef: new	26	
	\@gls@symbolsdef: new	26	
	General: Removed \acronymfont	129–132	
	\ACRfullfmt: new	198	
	\Acrfullfmt: new	197	
	\acrfullfmt: new	197	
	\ACRfullplfmt: new	199	
	\Acrfullplfmt: new	199	
	\acrfullplfmt: new	198	
	\acronymentry: new	201	
	sanitize: fixed bug that caused an error here	21	
	sc-short-long: new	204	
	sc-short-long-desc: new ...	206	
	\Genacrfullformat: new	90	
	\genacrfullformat: new	89	
	\GenericAcronymFields: new	201	
	\Genplacrfullformat: new ...	90	
	\genplacrfullformat: new ...	90	
	\Glsentryfull: bug fix: added missing \acronymfont	139	
	\glsentryfull: bug fix: added missing \acronymfont	139	
	\Glsentryfullpl: bug fix: added missing \acronymfont	139	
	\glsentryfullpl: bug fix: added missing \acronymfont	139	
	\glsngenacfmt: new	87	
	\GlsUseAcrEntryDispStyle: new	202	
	\GlsUseAcrStyleDefs: new ..	202	
	short-long: new	203	
	short-long-desc: new	206	
	xindynoglsnumbers: new	25	
	sm-short-long: new	204	
	sm-short-long-desc: new ...	206	
	\makeglossaries: made pream- ble only	153	
	index: new	27	
	\newacronymstyle: new	202	
	long-sc-short: new	204	
	long-sc-short-desc: new ...	205	
	long-short: new	203	
	long-short-desc: new	205	
	long-sm-short: new	204	
	long-sm-short-desc: new ...	205	
	footnote: new	209	
	footnote-desc: new	211	
	footnote-sc: new	210	
	footnote-sc-desc: new	211	
	footnote-sm: new	210	
	footnote-sm-desc: new	211	
	\setacronymstyle: new	201	
	\SetDescriptionAcronymDisplayStyle: Moved check for empty cus- tom text to prevent unwanted parenthetical material	219	
	\SetDescriptionFootnoteAcronymDisplayStyle: Moved check for empty cus- tom text to prevent unwanted parenthetical material	215	
	\SetFootnoteAcronymDisplayStyle: Moved check for empty cus- tom text to prevent unwanted parenthetical material	221	
	\SetGenericNewAcronym: new	200	
	\SetSmallAcronymDisplayStyle: Moved check for empty cus- tom text to prevent unwanted parenthetical material	224	
	dua: new	206	
	dua-desc: new	208	
	numberedsection: added		
	nameref option	6	
4.03			
	\@do@wrglossary: added		
	\glsdetoklabel	162	
	\@ACRlong: removed \glslabel (defined in \@gls@link) ...	329	
	\@ACRshort: removed \glslabel (defined in \@gls@link) ...	328	
	\@Acrlong: removed \glslabel (defined in \@gls@link) ...	329	
	\@Acrshort: removed \glslabel (defined in \@gls@link) ...	328	
	\@GLS@: removed \glslabel (de- fined in \@gls@link)	108	

<code>\@GLSpl:</code> removed <code>\glslabel</code> (defined in <code>\@gls@link</code>) ...	111	formation	327
<code>\@Gls@:</code> removed <code>\glslabel</code> (de- fined in <code>\@gls@link</code>)	107	<code>\genplacrfullformat:</code> rede- fined to use accessibility in- formation	327
<code>\@Gls@entry@field:</code> new	133	<code>\glossentryname:</code> added	
<code>\@GLspl@:</code> removed <code>\glslabel</code> (defined in <code>\@gls@link</code>) ...	110	<code>\glsdetoklabel</code>	186
<code>\@acrlong:</code> removed <code>\glslabel</code> (defined in <code>\@gls@link</code>) ...	328	<code>\gls@defglossaryentry:</code> added	
<code>\@acrsort:</code> removed <code>\glslabel</code> (defined in <code>\@gls@link</code>) ...	327	<code>\glsdetoklabel</code>	71
<code>\@gls@:</code> removed <code>\glslabel</code> (de- fined in <code>\@gls@link</code>)	106	replaced #1 with <code>\@glo@label</code>	72
<code>\@gls@access@display:</code> new ..	316	replaced <code>\ifthenelse</code> with <code>\ifdefequal</code>	73
<code>\@gls@entry@field:</code> new	133	<code>\glsadd:</code> added <code>\glsdetoklabel</code>	141
<code>\@gls@fetchfield:</code> new	67	<code>\glsaddkey:</code> switched to using <code>\@gls@field@link</code>	68
<code>\@gls@field@link:</code> new	112	<code>\glsdetoklabel:</code> new	50
<code>\@gls@link:</code> added <code>\glsdetoklabel</code>	95	<code>\glsdisplaynumberlist:</code> added <code>\glsdetoklabel</code>	140
moved <code>\@gls@link@opts</code> and <code>\@gls@link@label</code> to <code>\@gls@link</code>	95	<code>\glsdoifexistsorwarn:</code> new ..	51
<code>\@gls@writedef:</code> added <code>\glsdetoklabel</code>	66	<code>\glsentryaccess:</code> switched to using <code>\@gls@entry@field</code> ..	314
<code>\@glsdisp:</code> removed <code>\glslabel</code> (defined in <code>\@gls@link</code>) ...	111	<code>\glsentrydescaccess:</code> switched to using <code>\@gls@entry@field</code>	315
<code>\@GLspl@:</code> removed <code>\glslabel</code> (defined in <code>\@gls@link</code>) ...	109	<code>\glsentrydescpluralaccess:</code> switched to using <code>\@gls@entry@field</code>	315
<code>\@printglossary:</code> added <code>\glsdetoklabel</code>	168	<code>\glsentryfirstaccess:</code> switched to using <code>\@gls@entry@field</code>	314
General: changed default to <code>\@empty</code> instead of <code>\relax</code> ..	25	<code>\glsentryfirstplural:</code> added <code>\glsdetoklabel</code>	136
removed <code>\glslabel</code> (defined in <code>\@gls@link</code>)	126	<code>\glsentrylongaccess:</code> switched to using <code>\@gls@entry@field</code>	315
<code>sc-short-long-desc:</code> redefined to use accessibility informa- tion	333	<code>\glsentrylongpluralaccess:</code> switched to using <code>\@gls@entry@field</code>	315
<code>\compatibleglossentry:</code> added <code>\glsdetoklabel</code>	310	<code>\glsentrypluralaccess:</code> switched to using <code>\@gls@entry@field</code>	314
<code>\compatiblesubglossentry:</code> added <code>\glsdetoklabel</code> ...	310	<code>\glsentryshortaccess:</code> switched to using <code>\@gls@entry@field</code>	315
<code>\Genacrfullformat:</code> redefined to use accessibility informa- tion	327	<code>\glsentryshortpluralaccess:</code> switched to using <code>\@gls@entry@field</code>	315
<code>\genacrfullformat:</code> redefined to use accessibility informa- tion	326	<code>\glsentrysymbolaccess:</code> switched to using <code>\@gls@entry@field</code>	314
<code>\Genplacrfullformat:</code> rede- fined to use accessibility in-		<code>\glsentrysymbolpluralaccess:</code> switched to using <code>\@gls@entry@field</code>	315

<code>\glsentrytextaccess:</code>	switched to using <code>\@gls@entry@field</code>	314	<code>\ifglshasshort:</code>	added	
<code>\glsngenacfmt:</code>	redefined to use accessibility information	... 324	<code>\glsdetoklabel</code>	52
<code>\glsngenentryfmt:</code>	redefined to use accessibility information	322	<code>\ifglshassymbol:</code>	added	
<code>\gls hyperlink:</code>	added <code>\glsdetoklabel</code> 141	<code>\glsdetoklabel</code>	52
<code>\glslocalreset:</code>	added		replaced <code>\ifcsempy</code> with <code>\ifdefempty</code> and replaced <code>\ifx</code> with <code>\ifdefequal</code>	52
<code>\glsdetoklabel</code>	79	<code>\ifglssused:</code>	added <code>\glsdetoklabel</code> 50
<code>\glslocalunset:</code>	added		<code>sm-short-long-desc:</code>	redefined to use accessibility information 334
<code>\glsdetoklabel</code>	79	<code>long-sc-short-desc:</code>	redefined to use accessibility information 332
<code>\glsmoveentry:</code>	added <code>\glsdetoklabel</code> 76	<code>long-short:</code>	redefined to use accessibility information 331
replaced <code>\ifthenelse</code> with <code>\ifdefequal</code>	76	<code>long-short-desc:</code>	redefined to use accessibility information	332
<code>\glsrefentry:</code>	added <code>\glsdetoklabel</code> 184	<code>long-sm-short-desc:</code>	redefined to use accessibility information 333
<code>\glsreset:</code>	added <code>\glsdetoklabel</code> 79	<code>footnote:</code>	redefined to use accessibility information 337
<code>\glsseelist:</code>	added <code>\expandafter</code> commands 165	<code>footnote-desc:</code>	redefined to use accessibility information	... 339
<code>\glsstepentry:</code>	added <code>\glsdetoklabel</code> 184	<code>footnote-sc:</code>	redefined to use accessibility information 338
<code>\glsstepsubentry:</code>	added		<code>footnote-sc-desc:</code>	redefined to use accessibility information	339
<code>\glsdetoklabel</code>	184	<code>footnote-sm:</code>	redefined to use accessibility information 339
<code>\glsunset:</code>	added <code>\glsdetoklabel</code> 79	<code>footnote-sm-desc:</code>	redefined to use accessibility information	340
<code>short-long:</code>	commented spurious EOL 204	<code>\renewacronymstyle:</code>	new	... 202
redefined to use accessibility information	331	<code>\showglocounter:</code>	added	
<code>short-long-desc:</code>	redefined to use accessibility information	333	<code>\glsdetoklabel</code>	232
<code>\ifglsdscsuppressed:</code>	added		<code>\showglodesc:</code>	added <code>\glsdetoklabel</code> 233
<code>\glsdetoklabel</code>	52	fixed typo	52
<code>\ifglshaschildren:</code>	added		<code>\showglodescaccess:</code>	added	
<code>\glsdetoklabel</code>	51	<code>\glsdetoklabel</code>	346
<code>\ifglshasdesc:</code>	added <code>\glsdetoklabel</code> 52	<code>\showglodescplural:</code>	added	
<code>\ifglshasfield:</code>	new 53	<code>\glsdetoklabel</code>	234
<code>\ifglshaslong:</code>	added <code>\glsdetoklabel</code> 52	<code>\showglodescpluralaccess:</code>	added <code>\glsdetoklabel</code>	... 346
<code>\ifglshasparent:</code>	added		<code>\showglofirst:</code>	added <code>\glsdetoklabel</code> 232
<code>\glsdetoklabel</code>	51	<code>\showglofirstaccess:</code>	added	
			<code>\glsdetoklabel</code>	345

<code>\showglofirstpl:</code>	added	<code>\showglouserii:</code>	added
<code>\glsdetoklabel</code> 232	<code>\glsdetoklabel</code> 232
<code>\showglofirstpluralaccess:</code>		<code>\showglouseriii:</code>	added
added <code>\glsdetoklabel</code>	... 345	<code>\glsdetoklabel</code> 233
<code>\showgloflag:</code>	added <code>\glsdetoklabel</code>	<code>\showglouseriv:</code>	added
.....	235	<code>\glsdetoklabel</code> 233
<code>\showgloindex:</code>	added <code>\glsdetoklabel</code>	<code>\showglouseriv:</code>	added
.....	235	<code>\glsdetoklabel</code> 233
<code>\showglolevel:</code>	added <code>\glsdetoklabel</code>	<code>\showglouserv:</code>	added <code>\glsdetoklabel</code>
.....	231	233
<code>\showglolevel:</code>	added <code>\glsdetoklabel</code>	<code>\showglouservi:</code>	added
.....	234	<code>\glsdetoklabel</code> 233
<code>\showglolevelaccess:</code>	added	dua: fixed bug in <code>\acrfullfmt</code>	. 208
<code>\glsdetoklabel</code> 346	fixed bug in <code>\Acrfullplfmt</code>	. 208
<code>\showglolevelpluralaccess:</code>		fixed bug in <code>\acrfullplfmt</code>	. 208
added <code>\glsdetoklabel</code>	... 346	redefined to use accessibility in-	
<code>\showglolevelpluralaccess:</code>		formation 334
added <code>\glsdetoklabel</code> 233	dua-desc: commented spurious	
<code>\showglolevelpluralaccess:</code>		EOL 209
added <code>\glsdetoklabel</code> 345	redefined to use accessibility in-	
<code>\showglolevelpluralaccess:</code>		formation 336
added <code>\glsdetoklabel</code> 231		
<code>\showglolevelpluralaccess:</code>			
added <code>\glsdetoklabel</code> 231	<code>\@@gls@noidx@nosanitizesort:</code>	
<code>\showglolevelpluralaccess:</code>		new 19
added <code>\glsdetoklabel</code> 345	<code>\@@gls@noidx@sanitizesort:</code>	
<code>\showglolevelpluralaccess:</code>		new 19
added <code>\glsdetoklabel</code> 234	<code>\@@gls@nosanitizesort:new</code>	. 18
<code>\showglolevelpluralaccess:</code>		<code>\@@gls@sanitizesort:new</code>	... 18
added <code>\glsdetoklabel</code> 346	<code>\@glo@addchildren:new</code>	... 171
<code>\showglolevelpluralaccess:</code>		<code>\@glo@do@sortentries:new</code>	. 171
added <code>\glsdetoklabel</code> 346	<code>\@glo@grabfirst:new</code> 177
<code>\showglolevelpluralaccess:</code>		<code>\@glo@sortedinsert:new</code>	... 172
added <code>\glsdetoklabel</code> 234	<code>\@glo@sortentries:new</code>	... 170
<code>\showglolevelpluralaccess:</code>		<code>\@glo@sorthandler@case:new</code>	173
added <code>\glsdetoklabel</code> 234	<code>\@glo@sorthandler@letter:</code>	
<code>\showglolevelpluralaccess:</code>		new 173
added <code>\glsdetoklabel</code> 346	<code>\@glo@sorthandler@nocase:</code>	
<code>\showglolevelpluralaccess:</code>		new 173
added <code>\glsdetoklabel</code> 234	<code>\@glo@sorthandler@word:new</code>	172
<code>\showglolevelpluralaccess:</code>		<code>\@glo@sortmacro@case:new</code>	. 174
added <code>\glsdetoklabel</code>	... 346	<code>\@glo@sortmacro@def:new</code>	.. 175
<code>\showglolevelpluralaccess:</code>		<code>\@glo@sortmacro@def@do:new</code>	175
added <code>\glsdetoklabel</code> 231	<code>\@glo@sortmacro@letter:new</code>	174
<code>\showglolevelpluralaccess:</code>		<code>\@glo@sortmacro@nocase:new</code>	174
added <code>\glsdetoklabel</code> 345	<code>\@glo@sortmacro@standard:</code>	
<code>\showglolevelpluralaccess:</code>		new 174
added <code>\glsdetoklabel</code> 232	<code>\@glo@sortmacro@use:new</code>	.. 175
<code>\showglolevelpluralaccess:</code>		<code>\@glo@sortmacro@word:new</code>	. 173

<code>\@gls@getcounterprefix:</code>	<code>\printnoidxglossary:</code> new .. 167
added warning if no prefix can be formed	<code>\showgloclist:</code> new
164	235
<code>\@gls@getothergrouptitle:</code>	<code>\warn@noprintglossary:</code> Acti- vate warning in <code>\makeglossaries</code>
new	166
190	<code>\writeist:</code> checked for definition of <code>\glswrite</code>
<code>\@gls@noidx@do:</code> new	144, 148
177	4.06
<code>\@gls@noref@warn:</code> new	<code>\@GLS@:</code> added <code>\glsifhyper</code> .. 108
157	<code>\@GLSp1:</code> added <code>\glsifhyper</code> . 111
<code>\@gls@reference:</code> new	<code>\@Gls@:</code> added <code>\glsifhyper</code> .. 107
179	<code>\@Glspl@:</code> added <code>\glsifhyper</code> 110
<code>\@gls@warnonglossdefined:</code>	<code>\@gls@:</code> added <code>\glsifhyper</code> .. 107
new	<code>\@gls@numbersdef:</code> added hook to set toc title
17	27
<code>\@gls@warnontheglossdefined:</code>	<code>\@gls@symbolsdef:</code> added hook to set toc title
new	26
17	<code>\@glsdisp:</code> added <code>\glsifhyper</code> 111
<code>\@no@makeglossaries:</code> new .. 157	<code>\@glspl@:</code> added <code>\glsifhyper</code> 109
<code>\@print@glossary:</code> new	General: added <code>\glsifhyper</code>
169	126–132
<code>\@print@noidx@glossary:</code> new 175	acronym: added hook to set toc ti- tle
<code>\@printgloss@setsort:</code> new . 167	13
<code>\@printglossary:</code> new	acronyms: added hook to set toc title
167	14
General: added sort key to print- gloss group	<code>\glsdefmain:</code> added hook to set toc title
182	13
<code>\compatibleglossentry:</code>	4.07
changed <code>\newcommand</code> to <code>\def</code> as is may or may not be defined	<code>\@glossarysection:</code> added op- tional argument when using unstarred version
310	38
<code>\compatiblesubglossentry:</code>	<code>\@gls@noidx@do:</code> added <code>\global</code> in case it's used in a tabular- like style
changed <code>\newcommand</code> to <code>\def</code> as is may or may not be defined	177
310	<code>\Acrfullplfmt:</code> fixed no case change bug
<code>\defglsdisplayfirst:</code> fixed un- wanted space	<code>\glsletentryfield:</code> new
91	133
<code>\glo@grabfirst:</code> new	4.08
176	<code>\@ACRlong:</code> added <code>\do@gls@link@checkfirsthyper</code>
<code>\gls@defglossaryentry:</code> re- placed <code>\ifx</code> with <code>\ifdefvoid</code> 76	329
<code>\glsnoidxdisplayloc:</code> new .. 179	<code>\@ACRshort:</code> added <code>\do@gls@link@checkfirsthyper</code>
<code>\glsnoidxdisplayloclisthandler:</code>	328
new	<code>\@Acrlong:</code> added <code>\do@gls@link@checkfirsthyper</code>
179	329
<code>\glsnoidxloclist:</code> new	<code>\@Acrshort:</code> added <code>\do@gls@link@checkfirsthyper</code>
178	327
<code>\glsnoidxloclisthandler:</code>	<code>\@GLS@:</code> moved <code>\glsifhyper</code> .. 108 moved check for first use to <code>\@gls@link</code>
new	108
178	
<code>\glsnoidxstripaccents:</code> new . 19	
<code>alttree:</code> moved hangindent and parindent assignments out- side level test	
287	
<code>\makeglossaries:</code> Moved def- inition of <code>\glswrite</code> to <code>\makeglossaries</code>	
152	
<code>\makenoidxglossaries:</code> new . 154	
<code>\printglossary:</code> changed to use new <code>\@printglossary</code> ... 167	
<code>\printnoidxglossaries:</code> new 167	

\@GLSpl:moved \glsifhyper . 111	removed \@sacrlong 129
moved check for first use to	removed \@sACRshortpl . . . 128
\@gls@link 111	removed \@sAcrshortpl . . . 128
\@Gls@:moved \glsifhyper .. 107	removed \@sacrshortpl . . . 127
moved check for first use to	removed \@sACRshort 127
\@gls@link 107	removed \@sAcrshort 126
\@GLspl@:moved \glsifhyper 110	removed \@sacrshort 126
moved check for first use to	removed \@sgls@link 93
\@gls@link 110	removed \@sGLSdescplural 118
\@acrlong:added \do@gls@link@checkfirsthyper	removed \@sGLSdescplural 118
. 328	removed \@sglsdescplural 117
\@acrshort:added \do@gls@link@checkfirsthyper	removed \@sGLSdesc 117
. 327	removed \@sGLSdesc 117
\@closegls:new 150	removed \@sglsdesc 117
\@gls@:moved \glsifhyper .. 107	removed \@sglsdisp 111
moved check for first use to	removed \@sGLSfirstplural 115
\@gls@link 107	removed \@sGlsfirstplural 115
\@gls@doautomake:new 25	removed \@sglsfirstplural 115
\@gls@field@link: added as-	removed \@sGLSfirst 114
signment of \do@gls@link@checkfirsthyper	removed \@sGlsfirst 113
. 112	removed \@sglsfirst 113
\@gls@forbidtextext:new 54	removed \@sGLSname 116
\@gls@hyp@opt:new 93	removed \@sGlsname 116
\@gls@link: removed redun-	removed \@sglsname 116
dancy 95	removed \@sGLSplural 114
renamed \gls@type to	removed \@sGlsplural 114
\gls@type 95	removed \@sglsplural 114
\@gls@link@checkfirsthyper:	removed \@sGLSpl 110
new 94	removed \@sGlspl 109
\@glsdisp:moved \glsifhyper 111	removed \@sglspl 109
moved check for first use to	removed \@sGLSsymbolplural
\@gls@link 111 120
\@glspl@:moved \glsifhyper 109	removed \@sGlsymbolplural
moved check for first use to 120
\@gls@link 109	removed \@sglssymbolplural
\@ignored@glossaries:new .. 58 119
General: added entrycounter op-	removed \@sGLSsymbol 119
tion to printgloss family . 181	removed \@sGlsymbol 119
added nopostdot option to	removed \@sglssymbol 118
printgloss family 181	removed \@sGLStext 112
added subentrycounter option	removed \@sGlstext 113
to printgloss family 181	removed \@sglstext 112
explicitly initialise hyper key .. 92	removed \@sGLSuseriii . . . 123
moved \glsifhyper . . . 126–132	removed \@sGlsuseriii . . . 122
removed \@sACRlongpl . . . 132	removed \@sglsuseriii . . . 122
removed \@sAcrlongpl . . . 131	removed \@sGLSuserii 122
removed \@sacrlongpl . . . 131	removed \@sGlsuserii 121
removed \@sACRlong 130	removed \@sglsuserii 121
removed \@sAcrlong 129	removed \@sGLSuseriv 123

removed \sGlsuseriv	123	\ACRlongpl: switched to using	
removed \sglsuseriv	123	\@gls@hyp@opt	132
removed \sGLSuseri	121	\Acrlongpl: switched to using	
removed \sGlsuseri	120	\@gls@hyp@opt	131
removed \sglsuseri	120	\acrlongpl: switched to using	
removed \sGLSuservi	125	\@gls@hyp@opt	131
removed \sGlsuservi	125	\ACRshort: switched to using	
removed \sglsuservi	125	\@gls@hyp@opt	127
removed \sGLSuserv	124	\Acrshort: switched to using	
removed \sGlsuserv	124	\@gls@hyp@opt	126
removed \sGlsuserv	124	\acrshort: switched to using	
removed \sGLS	108	\@gls@hyp@opt	125
removed \sGls	107	\ACRshortpl: switched to using	
removed \sgls	106	\@gls@hyp@opt	128
removed \@thirdofthree (de-		\Acrshortpl: switched to using	
finied in kernel)	106	\@gls@hyp@opt	128
removed sPGLS	242	\acrshortpl: switched to using	
removed sPglS	240	\@gls@hyp@opt	127
removed spglS	239	\forallacronyms: new	48
removed sPGLSpl	242	\GLS: switched to using	
removed sPglSpl	241	\@gls@hyp@opt	108
removed spglSpl	240	\Gls: switched to using	
\ACRfull: removed \s@ACRfull	198	\@gls@hyp@opt	107
switched to using \@gls@hyp@opt		\gls: switched to using	
.....	198	\@gls@hyp@opt	106
\Acrfull: removed \s@Acrfull	197	\gls@defglossaryentry: added	
switched to using \@gls@hyp@opt		check for ignored glossary ...	72
.....	197	\gls@istfilebase: new	33
\acrfull: removed \s@acrfull	196	\glsaddkey: removed \sGLS@user@<key>	
switched to using \@gls@hyp@opt		69
.....	196	removed \sGls@user@<key>	69
\ACRfullpl: removed \s@ACRfullpl		removed \sgls@user@<key>	68
.....	199	switched to using \@gls@hyp@opt	
switched to using \@gls@hyp@opt		68, 69
.....	199	\GLSdesc: switched to using	
\Acrfullpl: removed \s@Acrfullpl		\@gls@hyp@opt	117
.....	198	\Glsdesc: switched to using	
switched to using \@gls@hyp@opt		\@gls@hyp@opt	117
.....	198	\glsdesc: switched to using	
\acrfullpl: removed \s@acrfullpl		\@gls@hyp@opt	116
.....	198	\GLSdescplural: switched to using	
switched to using \@gls@hyp@opt		\@gls@hyp@opt	118
.....	198	\Glsdescplural: switched to using	
\ACRlong: switched to using		\@gls@hyp@opt	118
\@gls@hyp@opt	130	\glsdescplural: switched to using	
\Acrlong: switched to using		\@gls@hyp@opt	117
\@gls@hyp@opt	129	\glsdisablehyper: added	
\acrlong: switched to using		\KV@glslink@hyperfalse to	
\@gls@hyp@opt	129	definition	105

<code>\glsdisp</code> : switched to using <code>\@gls@hyp@opt</code>	111	<code>\glsymbol</code> : switched to using <code>\@gls@hyp@opt</code>	118
<code>\glsdohyperlink</code> : new	105	<code>\GLSsymbolplural</code> : switched to using <code>\@gls@hyp@opt</code>	120
<code>\glsdohypertarget</code> : new	105	<code>\Glsymbolplural</code> : switched to using <code>\@gls@hyp@opt</code>	119
<code>\glsenablehyper</code> : added <code>\KV@glslink@hypertrue</code> to definition	106	<code>\glsymbolplural</code> : switched to using <code>\@gls@hyp@opt</code>	119
<code>\GLSfirst</code> : switched to using <code>\@gls@hyp@opt</code>	113	<code>\GLStext</code> : switched to using <code>\@gls@hyp@opt</code>	112
<code>\Glsfirst</code> : switched to using <code>\@gls@hyp@opt</code>	113	<code>\Glstext</code> : switched to using <code>\@gls@hyp@opt</code>	113
<code>\glsfirst</code> : switched to using <code>\@gls@hyp@opt</code>	113	<code>\glstext</code> : switched to using <code>\@gls@hyp@opt</code>	112
<code>\GLSfirstplural</code> : switched to using <code>\@gls@hyp@opt</code>	115	<code>\glstreenamfmt</code> : new	282
<code>\Glsfirstplural</code> : switched to using <code>\@gls@hyp@opt</code>	115	<code>\GLSuseri</code> : switched to using <code>\@gls@hyp@opt</code>	121
<code>\glsfirstplural</code> : switched to using <code>\@gls@hyp@opt</code>	115	<code>\Glsuseri</code> : switched to using <code>\@gls@hyp@opt</code>	120
<code>\glsifhyper</code> : deprecated	93	<code>\glsuseri</code> : switched to using <code>\@gls@hyp@opt</code>	120
<code>\glslink</code> : switched to using <code>\@gls@hyp@opt</code>	93	<code>\GLSuserii</code> : switched to using <code>\@gls@hyp@opt</code>	122
<code>\glslinkcheckfirsthyperhook</code> : new	94	<code>\Glsuserii</code> : switched to using <code>\@gls@hyp@opt</code>	121
<code>\glslinkvar</code> : new	93	<code>\glsuserii</code> : switched to using <code>\@gls@hyp@opt</code>	121
<code>\GLSname</code> : switched to using <code>\@gls@hyp@opt</code>	116	<code>\GLSuseriii</code> : switched to using <code>\@gls@hyp@opt</code>	122
<code>\Glsname</code> : switched to using <code>\@gls@hyp@opt</code>	116	<code>\Glsuseriii</code> : switched to using <code>\@gls@hyp@opt</code>	122
<code>\glsname</code> : switched to using <code>\@gls@hyp@opt</code>	116	<code>\glsuseriii</code> : switched to using <code>\@gls@hyp@opt</code>	122
<code>\GLSpl</code> : switched to using <code>\@gls@hyp@opt</code>	110	<code>\GLSuseriv</code> : switched to using <code>\@gls@hyp@opt</code>	123
<code>\Glspl</code> : switched to using <code>\@gls@hyp@opt</code>	109	<code>\Glsuseriv</code> : switched to using <code>\@gls@hyp@opt</code>	123
<code>\glspl</code> : switched to using <code>\@gls@hyp@opt</code>	109	<code>\glsuseriv</code> : switched to using <code>\@gls@hyp@opt</code>	123
<code>\GLSplural</code> : switched to using <code>\@gls@hyp@opt</code>	114	<code>\GLSuserv</code> : switched to using <code>\@gls@hyp@opt</code>	124
<code>\Glsplural</code> : switched to using <code>\@gls@hyp@opt</code>	114	<code>\Glsuserv</code> : switched to using <code>\@gls@hyp@opt</code>	124
<code>\glsplural</code> : switched to using <code>\@gls@hyp@opt</code>	114	<code>\glsuserv</code> : switched to using <code>\@gls@hyp@opt</code>	124
<code>\glspace</code> : new	197	<code>\GLSuservi</code> : switched to using <code>\@gls@hyp@opt</code>	125
<code>\GLSsymbol</code> : switched to using <code>\@gls@hyp@opt</code>	119	<code>\Glsuservi</code> : switched to using <code>\@gls@hyp@opt</code>	125
<code>\Glsymbol</code> : switched to using <code>\@gls@hyp@opt</code>	119		

\glsuservi: switched to using	
\@gls@hyp@opt	125
\ifignoredglossary:new	58
altlongragged4col: fixed bug	
that displayed description in-	
stead of symbol	264
\newglossary: added starred ver-	
sion	55
\newignoredglossary:new ...	57
\ns@newglossary: added	
\@glotype@<name>@log ...	56
new	55
\p@gls@hyp@opt:new	93
\PGLS: changed to use	
\@gls@hyp@opt	242
\PglS: changed to use	
\@gls@hyp@opt	240
\pgls: changed to use	
\@gls@hyp@opt	239
\PGLSpl: changed to use	
\@gls@hyp@opt	242
\PglSpl: changed to use	
\@gls@hyp@opt	241
\pglSpl: changed to use	
\@gls@hyp@opt	240
\s@gls@hyp@opt:new	93
\s@newglossary:new	55
automake:new	25
4.09	
\glsaddkey: fixed bug in user	
commands	68
4.10	
\@Gls@acentryname:new ...	134
\@Gls@entryname:new	134
\@gls@glossary: Renamed	
\@glossary to \@gls@glossary	
.....	159
\glspercentchar:new	143
\glstildechar:new	143
almtree: moved space after sym-	
bol	287, 289
4.11	
\@do@wrglossary: added hook	162
sanitize: none option	21
\gls@wrglossary: renamed	
from \@wrglossary to	
\gls@wrglossary	159
\glsaddprotectedpagefmt:	
new	161
\glsbackslash:new	143

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.

Symbols	
<code>\@do@wrglossary</code>	162
<code>\@do@wrglossary</code>	161
<code>\@glo@assign@sortkey</code>	182
<code>\@glossarysec</code>	6
<code>\@glossaryseclabel</code>	6
<code>\@glossarysecstar</code>	6
<code>\@gls@default@entryfmt</code>	318
<code>\@gls@expand@field</code>	63
<code>\@gls@fixbraces</code>	165
<code>\@gls@noidx@nosanitizesort</code> .	19
<code>\@gls@noidx@sanitizesort</code> ...	19
<code>\@gls@nosanitizesort</code>	18
<code>\@gls@sanitizesort</code>	18
<code>\@ACRlong</code>	329
<code>\@ACRshort</code>	328
<code>\@Acrlong</code>	329
<code>\@Acrshort</code>	327
<code>\@GLS@</code>	108
<code>\@GLSpl</code>	110
<code>\@Gls@</code>	107
<code>\@Gls@acrenryname</code>	134
<code>\@Gls@entry@field</code>	133
<code>\@Gls@entryname</code>	134
<code>\@Glspl@</code>	110
<code>\@PGLS</code>	242
<code>\@PGLS@</code>	242
<code>\@PGLSpl</code>	242
<code>\@PGLSpl@</code>	243
<code>\@PglS</code>	240
<code>\@PglS@</code>	241
<code>\@PglSpl</code>	241
<code>\@PglSpl@</code>	241
<code>\@acrfull</code>	197
<code>\@acrlong</code>	328
<code>\@acrshort</code>	327
<code>\@addtoacronymlists</code>	14
<code>\@closegls</code>	150
<code>\@delimN</code>	193
<code>\@delimR</code>	193
<code>\@disable@onlypremakeg</code>	29
<code>\@disable@premakecs</code>	29
<code>\@disabled@glsaddxdycounters</code>	41
<code>\@do@seeglossary</code>	164
<code>\@do@wrglossary</code>	160, 291
<code>\@glo@addchildren</code>	171
<code>\@glo@default@sorttype</code>	10
<code>\@glo@do@sortentries</code>	171
<code>\@glo@grabfirst</code>	177
<code>\@glo@no@assign@sortkey</code>	182
<code>\@glo@seeautonumberlist</code>	8
<code>\@glo@sortedinsert</code>	172
<code>\@glo@sortentries</code>	170
<code>\@glo@sorthandler@case</code>	173
<code>\@glo@sorthandler@letter</code> ...	173
<code>\@glo@sorthandler@nocase</code> ...	173
<code>\@glo@sorthandler@word</code>	172
<code>\@glo@sortmacro@case</code>	174
<code>\@glo@sortmacro@def</code>	175
<code>\@glo@sortmacro@def@do</code>	175
<code>\@glo@sortmacro@letter</code>	174
<code>\@glo@sortmacro@nocase</code>	174
<code>\@glo@sortmacro@standard</code> ...	174
<code>\@glo@sortmacro@use</code>	175
<code>\@glo@sortmacro@word</code>	173
<code>\@glo@storeentry</code>	77
<code>\@glo@types</code>	54
<code>\@glossary@default@style</code>	7
<code>\@glossaryentryfield</code>	76
<code>\@glossarysection</code>	38
<code>\@glossarysubentryfield</code>	77
<code>\@gls</code>	106
<code>\@gls@</code>	106
<code>\@gls@@link</code>	94
<code>\@gls@access@display</code>	316
<code>\@gls@addpredefinedattributes</code>	42
<code>\@gls@assign@symbol@field</code> ...	18
<code>\@gls@assign@symbolplural@field</code>	18
<code>\@gls@checkactual</code>	103
<code>\@gls@checkbar</code>	102
<code>\@gls@checkescactual</code>	100
<code>\@gls@checkescbar</code>	101
<code>\@gls@checkesclevel</code>	101
<code>\@gls@checkescquote</code>	100

<code>\@gls@checklevel</code>	102	<code>\@gls@sanitizesort</code>	18
<code>\@gls@checkmkidxchars</code>	98	<code>\@gls@sanitizesymbol</code>	18
<code>\@gls@checkquote</code>	99	<code>\@gls@saveentrycounter</code>	96
<code>\@gls@codepage</code>	47	<code>\@gls@setacrstyle</code>	23
<code>\@gls@counterwithin</code>	10	<code>\@gls@setcounter</code>	57
<code>\@gls@declareoption</code>	7	<code>\@gls@setupsort@def</code>	11
<code>\@gls@default@value</code>	59	<code>\@gls@setupsort@standard</code>	10
<code>\@gls@do@acronymsdef</code>	14	<code>\@gls@setupsort@use</code>	12
<code>\@gls@doautomake</code>	25	<code>\@gls@startswithexpandonce</code> ..	64
<code>\@gls@entry@field</code>	133	<code>\@gls@symbolsdef</code>	26
<code>\@gls@escbsdq</code>	98	<code>\@gls@tmpb</code>	99
<code>\@gls@expand@fields</code>	64	<code>\@gls@toc</code>	39
<code>\@gls@fetchfield</code>	67	<code>\@gls@updatechecked</code>	99
<code>\@gls@field@link</code>	112	<code>\@gls@warnonglossdefined</code>	17
<code>\@gls@fixbraces</code>	164	<code>\@gls@warnontheGLOSSdefined</code> .	17
<code>\@gls@forbidtext</code>	54	<code>\@gls@writedef</code>	66
<code>\@gls@getcounter</code>	57	<code>\@gls@xdy@Lclass@Alpha-page-numbers</code>	
<code>\@gls@getcounterprefix</code>	163	44
<code>\@gls@getgrouptitle</code>	189	<code>\@gls@xdy@Lclass@Appendix-page-numbers</code>	
<code>\@gls@getothergrouptitle</code> ...	190	44
<code>\@gls@glossary</code>	159	<code>\@gls@xdy@Lclass@Roman-page-numbers</code>	
<code>\@gls@hyp@opt</code>	93	44
<code>\@gls@hypergroup</code>	247	<code>\@gls@xdy@Lclass@alpha-page-numbers</code>	
<code>\@gls@ifinlist</code>	40	44
<code>\@gls@indexdef</code>	27	<code>\@gls@xdy@Lclass@arabic-page-numbers</code>	
<code>\@gls@keymap</code>	66, 237	44
<code>\@gls@link</code>	94	<code>\@gls@xdy@Lclass@arabic-section-numbers</code>	
<code>\@gls@link@checkfirsthyper</code> ..	94	44
<code>\@gls@loadlist</code>	8	<code>\@gls@xdy@Lclass@roman-page-numbers</code>	
<code>\@gls@loadlong</code>	8	43
<code>\@gls@loadsuper</code>	8	<code>\@gls@xdy@locationlist</code>	43
<code>\@gls@loadtree</code>	8	<code>\@gls@xdy@checkbackslash</code>	104
<code>\@gls@makefirststuc</code>	244	<code>\@gls@xdy@checkquote</code>	103
<code>\@gls@missingnumberlist</code>	62	<code>\@gls@Alphacompositor</code>	34, 44
<code>\@gls@noaccess</code>	312	<code>\@gls@acronymlists</code>	14
<code>\@gls@noexpand@field</code>	63	<code>\@gls@defaultplural</code>	62
<code>\@gls@noexpand@fields</code>	63	<code>\@gls@defaultsort</code>	63
<code>\@gls@nohyperlist</code>	16	<code>\@gls@disp</code>	111
<code>\@gls@noidx@do</code>	177	<code>\@gls@firstletter</code>	143
<code>\@gls@noidx@setsanitizesort</code> .	21	<code>\@gls@hypernumber</code>	193
<code>\@gls@noref@warn</code>	157	<code>\@gls@link</code>	105
<code>\@gls@notranslatorhook</code>	22	<code>\@gls@minrange</code>	143
<code>\@gls@numbersdef</code>	26	<code>\@gls@nextpages</code>	183
<code>\@gls@onlypremakeg</code>	29	<code>\@gls@nodesc</code>	62
<code>\@gls@provide@newglossary</code> ...	54	<code>\@gls@noname</code>	62
<code>\@gls@reference</code>	179	<code>\@gls@nonextpages</code>	182
<code>\@gls@renewglossary</code>	159	<code>\@gls@openfile</code>	150
<code>\@gls@sanitizedesc</code>	17	<code>\@gls@order</code>	24
<code>\@gls@sanitizename</code>	18	<code>\@gls@spl@</code>	109

<code>\@glstarget</code>	105	<code>\Acfp</code>	213
<code>\@glswidestname</code>	287	<code>\acfp</code>	212
<code>\@glswritefiles</code>	158	<code>\Acl</code>	212
<code>\@ignored@glossaries</code>	58	<code>\acl</code>	212
<code>\@istfilename</code>	33	<code>\Aclp</code>	212
<code>\@makeglossary</code>	150	<code>\aclp</code>	212
<code>\@mfu@nocaplist</code>	245	<code>\Acp</code>	213
<code>\@newglossary</code>	57	<code>\acp</code>	213
<code>\@newglossaryentryposthook</code>	76	<code>\acrfootnote</code>	215
<code>\@newglossaryentryprehook</code>	76	<code>\ACRfull</code>	198
<code>\@no@makeglossaries</code>	157	<code>\Acrfull</code>	197
<code>\@no@post@desc</code>	33	<code>\acrfull</code>	196, 197, 201, 209, 337
<code>\@nopostdesc</code>	32	<code>\ACRfullfmt</code>	198
<code>\@onlypremakeg</code>	29	<code>\Acrfullfmt</code>	197
<code>\@p@glossarysection</code>	38	<code>\acrfullfmt</code>	197
<code>\@p@gl</code>	239	<code>\acrfullformat</code>	87, 197
<code>\@p@gl@</code>	239	<code>\ACRfullpl</code>	199
<code>\@p@gl@pl</code>	240	<code>\Acrfullpl</code>	198
<code>\@p@gl@spl@</code>	240	<code>\acrfullpl</code>	198
<code>\@print@glossary</code>	169	<code>\ACRfullplfmt</code>	199
<code>\@print@noidx@glossary</code>	175	<code>\Acrfullplfmt</code>	199
<code>\@print@gloss@set@sort</code>	167	<code>\acrfullplfmt</code>	198
<code>\@print@glossary</code>	167	<code>\acrlinkfootnote</code>	215
<code>\@set@glo@num@format</code>	97, 292	<code>\acrlinkfullformat</code>	197
<code>\@wrglossary@page@format</code>	161	<code>\ACRlong</code>	130
<code>\@wrglossary@number@hook</code>	161	<code>\Acrlong</code>	129
<code>\@xdy@main@language</code>	24	<code>\acrlong</code>	129
<code>\@xdy@attributelist</code>	40	<code>\ACRlongpl</code>	132
<code>\@xdy@attributes</code>	39	<code>\Acrlongpl</code>	131
<code>\@xdy@language</code>	47	<code>\acrlongpl</code>	131
<code>\@xdy@letter@groups</code>	48	<code>\acrnameformat</code>	199, 221
<code>\@xdy@location@class@order</code>	45	<code>\acrno@link@footnote</code>	215
<code>\@xdy@loc@ref</code>	40	acronym (option)	13
<code>\@xdy@required@styles</code>	46	acronym styles:	
<code>\@xdy@sort@rules</code>	46	<code>dua</code>	206, 334
<code>\@xdy@user@alphabets</code>	42	<code>dua-desc</code>	208, 336
<code>\@xdy@user@location@defs</code>	44	<code>footnote</code>	209, 337
<code>\@xdy@user@location@names</code>	44	<code>footnote-desc</code>	211, 339
A			
<code>\Ac</code>	213	<code>footnote-sc</code>	210, 338
<code>\ac</code>	213	<code>footnote-sc-desc</code>	211, 339
<code>access (key)</code>	311	<code>footnote-sm</code>	210, 339
<code>accsupp package</code>	310	<code>footnote-sm-desc</code>	211, 340
<code>\accsuppglossaryentryfield</code>	330	<code>long-sc-short</code>	204
<code>\accsuppglossarysubentryfield</code>	330	<code>long-sc-short-desc</code>	205, 332
.....	330	<code>long-short</code>	203, 331
<code>\Acf</code>	212	<code>long-short-desc</code>	205, 332
<code>\acf</code>	212	<code>long-sm-short</code>	204
		<code>long-sm-short-desc</code>	205, 333
		<code>sc-short-long</code>	204

sc-short-long-desc .. 206, 333
 short-long 203, 331
 short-long-desc 206, 333
 sm-short-long 204
 sm-short-long-desc .. 206, 334
 \acronymentry 201
 \acronymfont
 87, 199, 217, 221, 223, 226
 acronymlists (option) 16
 \acronymname 30
 acronyms (option) 14
 \acronymsort 201
 \acronymtype 13, 196
 \acrpluralsuffix 196
 \ACRshort 127
 \Acrshort 126
 \acrshort 125
 \ACRshortpl 128
 \Acrshortpl 128
 \acrshortpl 127
 \Acs 212
 \acs 212
 \Acspace 212
 \acspace 212
 \addglossarytocaptions 31
 \addto 31
 align (environment) 78, 96
 altlist (style) 252
 altlistgroup (style) 253
 altlisthypergroup (style) 253
 altlong4col (style) 259
 altlong4colborder (style) 259
 altlong4colheader (style) 259
 altlong4colheaderborder (style) 259
 altlongragged4col (style) 263
 altlongragged4colborder (style) 265
 altlongragged4colheader (style) 264
 altlongragged4colheaderborder
 (style) 265
 \altnewglossary 56
 altsuper4col (style) 275
 altsuper4colborder (style) 275
 altsuper4colheader (style) 275
 altsuper4colheaderborder
 (style) 275
 altsuperragged4col (style) 280
 altsuperragged4colborder
 (style) 281

altsuperragged4colheader
 (style) 281
 altsuperragged4colheaderborder
 (style) 281
 alttree (style) 287
 alttreegroup (style) 289
 alttreehypergroup (style) 289
 amsgen package 5, 92
 amsmath package 78
 \andname 31
 array package 260, 276
 article class 163
 automake (option) 25

B

babel package 28, 30, 31, 47, 347

C

\capitalisewords 245
 \compatglossarystyle 296
 compatible-2.07 (option) 26
 compatible-3.07 (option) 26
 \compatibleglossentry .. 186, 310
 \compatiblesubglossentry 187, 310
 counter (key) 60
 counter (option) 16
 \CustomAcronymFields 229
 \CustomNewAcronymDef 229

D

datatool package 172
 \DeclareAcronymList 15
 \DefaultNewAcronymDef .. 213, 340
 \defentryfmt 91
 \defglstdisplay 90
 \defglstdisplayfirst 91
 \defglstentry 56
 \defglstentryfmt 54, 58, 60, 82
 \DefineAcronymSynonyms 211
 \delimN 35, 192
 \delimR 35, 192
 description (environment) 251
 description (key) 58
 description (option) 23
 descriptionaccess (key) 311
 \DescriptionDUANewAcronymDef 217
 \DescriptionFootnoteNewAcronymDef
 216, 341
 \descriptionname 30

- \DescriptionNewAcronymDef ..
..... 220, 342
- descriptionplural (key) 59
- descriptionpluralaccess (key) 312
- \detokenize 50
- doc package 5, 12
- dua (acrstyle) 206, 334
- dua (option) 23
- dua-desc (acrstyle) 208, 336
- \DUANewAcronymDef 227

E

- entrycounter (option) 9
- entrycounterwithin (option) ... 10
- \entryname 30
- environments:
 - align 78, 96
 - description 251
 - longtable 8, 230, 254–265
 - multicols 266
 - supertabular ... 8, 231, 269–282
 - theglossary ... 5, 17, 35, 36,
185, 186, 191, 267, 268, 284–287
 - theindex 282
- equation (counter) 96, 97
- etoolbox package 5, 243

F

- file types
 - .aux 169
 - .glo 77
 - .ist 142, 143, 149, 150
 - .toc 39
 - .xdy 33
 - glo 236
- \findrootlanguage 47
- first (key) 59
- firstaccess (key) 311
- \firstacronymfont 87, 199
- firstplural (key) 59
- firstpluralaccess (key) 311
- footnote (acrstyle) 209, 337
- footnote (option) 23
- footnote-desc (acrstyle) .. 211, 339
- footnote-sc (acrstyle) 210, 338
- footnote-sc-desc (acrstyle) 211, 339
- footnote-sm (acrstyle) 210, 339
- footnote-sm-desc (acrstyle) 211, 340
- \FootnoteNewAcronymDef . 222, 343
- \foralllacronyms 48

- \foralllglossaries 48
- \foralllglsentries 49
- \forallglsentries 49

G

- garamondx package 196
- \Genacrfullformat 90, 327
- \genacrfullformat 89, 326
- \GenericAcronymFields 201
- \Genplacrfullformat 90, 327
- \genplacrfullformat 90, 327
- \glo@grabfirst 176
- \glolinkprefix 96
- glossareentry (counter) 184
- glossaries package 27,
47, 144, 230, 237, 251, 290, 310
- glossaries-accsupp package . 76, 77, 310
- \GlossariesWarning 16
- \GlossariesWarningNoLine 16
- \glossary 55, 150, 159, 190
- glossary counters:
 - glossaryentry 183
 - glossarysubentry 183
- glossary keys:
 - access 311
 - counter 60
 - description 58
 - descriptionaccess 311
 - descriptionplural 59
 - descriptionpluralaccess . 312
 - first 59
 - firstaccess 311
 - firstplural 59
 - firstpluralaccess 311
 - long 62
 - longaccess 312
 - longplural 62
 - longpluralaccess 312
 - name 58
 - nonnumberlist 61
 - parent 61
 - plural 59
 - pluralaccess 311
 - see 60
 - short 62
 - shortaccess 312
 - shortplural 62
 - shortpluralaccess 312
 - sort 59

symbol	59
symbolaccess	311
symbolplural	60
symbolpluralaccess	311
text	59
textaccess	311
type	60
user1	61
user2	61
user3	61
user4	61
user5	61
user6	61
glossary package	1, 195
glossary styles:	
altlist	252, 253, 298
altlist	252
altlistgroup	253, 298
altlistgroup	253
altlisthypergroup	253, 298
altlisthypergroup	253
altlong4col	259, 263, 300
altlong4col	259
altlong4colborder	259, 300
altlong4colborder	259
altlong4colheader	259, 300
altlong4colheader	259
altlong4colheaderborder	259, 300
altlong4colheaderborder	259
altlongragged4col	263–265, 301
altlongragged4col	263
altlongragged4colborder	265, 302
altlongragged4colborder	265
altlongragged4colheader	264, 301
altlongragged4colheader	264
altlongragged4colheaderborder	265, 302
altlongragged4colheaderborder	265
altsuper4col	275, 280, 309
altsuper4col	275
altsuper4colborder	275, 309
altsuper4colborder	275
altsuper4colheader	275, 309
altsuper4colheader	275
altsuper4colheaderborder	275, 309
altsuper4colheaderborder	275
altsuperragged4col	280, 281, 307
altsuperragged4col	280
altsuperragged4colborder	281, 307
altsuperragged4colborder	281
altsuperragged4colheader	281, 307
altsuperragged4colheader	281
altsuperragged4colheaderborder	281, 308
altsuperragged4colheaderborder	281
alttree	268, 287, 289, 304
alttree	287
alttreegroup	289, 305
alttreegroup	289
alttreehypergroup	289, 305
alttreehypergroup	289
index	266, 282–284, 302
index	282
indexgroup	283, 302
indexgroup	283
indexhypergroup	283, 302
indexhypergroup	283
inline	297
inline	249
list	7, 251–253, 297
list	251
listdotted	253, 254, 298
listdotted	253
listgroup	252, 297
listgroup	252
listhypergroup	252, 298
listhypergroup	252
long	254–256, 260, 298, 300
long	254
long3col	256, 257, 299
long3col	256
long3colborder	256, 299
long3colborder	256
long3colheader	257, 299
long3colheader	257
long3colheaderborder	257, 299
long3colheaderborder	257
long4col	257–259, 299
long4col	257

long4colborder	258, 300
long4colborder	258
long4colheader	258, 300
long4colheader	258
long4colheaderborder		258, 300
long4colheaderborder	258
longborder	255, 299
longborder	255
longheader	255, 299
longheader	255
longheaderborder	255, 299
longheaderborder	255
longragged	260–262
longragged	260
longragged3col	... 262, 263, 301	
longragged3col	262
longragged3colborder		263, 301
longragged3colborder	263
longragged3colheader		263, 301
longragged3colheader	263
longragged3colheaderborder		
	263, 301
longragged3colheaderborder		
	263
longraggedborder	261, 301
longraggedborder	261
longraggedheader	261, 301
longraggedheader	261
longraggedheaderborder		262, 301
longraggedheaderborder	..	262
mcolalttree	268, 306
mcolalttree	268
mcolalttreegroup	269, 306
mcolalttreegroup	268
mcolalttreehypergroup		269, 306
mcolalttreehypergroup	... 269	
mcolindex	266, 305
mcolindex	266
mcolindexgroup	266, 305
mcolindexgroup	266
mcolindexhypergroup	.	266, 305
mcolindexhypergroup	266
mcoltree	267, 305
mcoltree	266
mcoltreegroup	305
mcoltreegroup	267
mcoltreehypergroup	..	267, 306
mcoltreehypergroup	267
mcoltreenoname	267, 306
mcoltreenoname	267
mcoltreenonamegroup	.	268, 306
mcoltreenonamegroup	267
mcoltreenonamehypergroup		
	268, 306
mcoltreenonamehypergroup		268
sublistdotted	298
sublistdotted	254
super	269–271, 277, 308
super	269
super3col	271, 272, 308
super3col	271
super3colborder	272, 308
super3colborder	272
super3colheader	272, 308
super3colheader	272
super3colheaderborder		272, 309
super3colheaderborder	...	272
super4col	273–275, 309
super4col	273
super4colborder	274, 309
super4colborder	274
super4colheader	274, 309
super4colheader	274
super4colheaderborder		274, 309
super4colheaderborder	...	274
superborder	270, 308
superborder	270
superheader	270, 308
superheader	270
superheaderborder	...	271, 308
superheaderborder	271
superragged	276, 278, 306
superragged	276
superragged3col	..	278–280, 307
superragged3col	278
superragged3colborder		279, 307
superragged3colborder	...	279
superragged3colheader		279, 307
superragged3colheader	...	279
superragged3colheaderborder		
	280, 307
superraggedborder	...	277, 306
superraggedborder	277
superraggedheader	...	277, 307
superraggedheader	277
superraggedheaderborder	.	
	278, 307
superraggedheaderborder	.	278

superraggedright3colheaderborder	\Gls	107, 109, 243
.....	\gls	5, 60,
tree		81, 92, 106, 108, 112–125, 184, 239
tree	\gls@Alphpage	160
treegroup	\gls@alphpage	160
treegroup	\gls@assign@desc	70
treehypergroup	\gls@assign@desc@field	18
treehypergroup	\gls@assign@descplural@field	18
treenoname	\gls@assign@field	64
treenoname	\gls@assign@name@field	18
treenonamegroup	\gls@assign@type@field	18
treenonamegroup	\gls@checkisacronymlist	15
treenonamehypergroup	\gls@checkseeallowed	60
treenonamehypergroup	\gls@codepage	24
glossary-hypernav package	\gls@defglossaryentry	70
glossary-list package	\gls@disablepagerefexpansion	160
glossary-long package	\gls@doclearpage	38
glossary-longragged package	\gls@doentryfmt	54
glossary-mcols package	\gls@glossary	159
glossary-super package	\gls@hypergroupprerun	247
.....	\gls@ifnotmeasuring	78
glossary-superragged package	\gls@istfilebase	33
glossary-tree package	\gls@level	63
glossaryentry (counter)	\gls@noidxglossary	157
.....	\gls@numberpage	161
glossaryentry (counter)	\gls@protected@pagefmts	160
\glossaryentryfield	\gls@Romanpage	161
\glossaryentrynumber	\gls@romanpage	161
\glossaryentrynumbers	\gls@save@numberlist	166
.....	\gls@suffixF	34
\glossaryheader	\gls@suffixFF	35
\glossarymark	\gls@wrglossary	159
\glossaryname	\glsaccessdisplay	317
\glossarypostamble	\glsaccsupp	315
\glossarypreamble	\glsadd	81, 141, 190
\glossarysection	\glsadd options	
\glossarystyle	counter	141
glossarysubentry (counter)	format	141, 192
.....	\glsaddall	50, 81, 142
glossarysubentry (counter)	\glsaddall options	
\glossarysubentryfield	types	141, 142
\glossentry	\glsaddallunused	142
\Glossentrydesc	\glsaddkey	67
\glossentrydesc	\GlsAddLetterGroup	48
\Glossentryname	\glsaddprotectedpagefmt	161
\glossentryname	\GlsAddSortRule	46
\Glossentrysymbol	\GlsAddXdyAlphabet	43
\glossentrysymbol	\GlsAddXdyAttribute	41, 290
\GLS	\GlsAddXdyCounters	40, 290

<code>\GlsAddXdyLocation</code>	45, 291	<code>\glsentryfmt</code>	58, 60, 81
<code>\GlsAddXdyStyle</code>	46	<code>\Glsentryfull</code>	139
<code>\glsautoprefix</code>	6	<code>\glsentryfull</code>	139, 201, 210, 338
<code>\glsbackslash</code>	143	<code>\Glsentryfullpl</code>	139
<code>\glsclearpage</code>	39	<code>\glsentryfullpl</code>	139
<code>\glsclosebrace</code>	143	<code>\glsentryitem</code>	185
<code>\glscompositor</code>	34, 44	<code>\Glsentrylong</code>	139
<code>\glscounter</code>	16, 56	<code>\glsentrylong</code>	139
<code>\GlsDeclareNoHyperList</code>	16	<code>\glsentrylongaccess</code>	315
<code>\glsdefaultttype</code>	13	<code>\Glsentrylongpl</code>	139
<code>\glsdefmain</code>	12	<code>\glsentrylongpl</code>	139
<code>\GLSdesc</code>	117	<code>\glsentrylongpluralaccess</code>	315
<code>\Glsdesc</code>	117	<code>\Glsentryname</code>	134
<code>\glsdesc</code>	116, 117	<code>\glsentryname</code>	133, 166
<code>\GLSdescplural</code>	118	<code>\glsentrynumberlist</code>	140, 155
<code>\Glsdescplural</code>	118	<code>\Glsentryplural</code>	135
<code>\glsdescplural</code>	117, 118	<code>\glsentryplural</code>	135
<code>\glsdescriptionaccessdisplay</code>	317	<code>\glsentrypluralaccess</code>	314
<code>\glsdescriptionpluralaccessdisplay</code>	317	<code>\Glsentryprefix</code>	238
<code>\glsdescwidth</code>	254, 260, 269, 276	<code>\glsentryprefix</code>	238
<code>\glsdetoklabel</code>	50	<code>\Glsentryprefixfirst</code>	238
<code>\glsdisablehyper</code>	105	<code>\glsentryprefixfirst</code>	238
<code>\glsdisp</code>	111	<code>\Glsentryprefixfirstplural</code>	238
<code>\glsdisplay</code>	81, 90, 106	<code>\glsentryprefixfirstplural</code>	238
<code>\glsdisplayfirst</code>	81, 90, 106	<code>\Glsentryprefixplural</code>	238
<code>\glsdisplaynumberlist</code>	140, 155, 179	<code>\glsentryprefixplural</code>	238
<code>\glsdohyperlink</code>	105	<code>\Glsentryshort</code>	138
<code>\glsdohypertarget</code>	105	<code>\glsentryshort</code>	138
<code>\glsdoifexists</code>	50	<code>\glsentryshortaccess</code>	315
<code>\glsdoifexistsorwarn</code>	51	<code>\Glsentryshortpl</code>	139
<code>\glsdoifnoexists</code>	50	<code>\glsentryshortpl</code>	138
<code>\glsdoparenifnotempty</code>	224	<code>\glsentryshortpluralaccess</code>	315
<code>\glsenablehyper</code>	106	<code>\glsentrysort</code>	137
<code>\glsentryaccess</code>	314	<code>\Glsentrysymbol</code>	136
<code>\glsentrycounter</code>	96	<code>\glsentrysymbol</code>	136
<code>\glsentrycounterlabel</code>	185	<code>\glsentrysymbolaccess</code>	314
<code>\Glsentrydesc</code>	135	<code>\Glsentrysymbolplural</code>	136
<code>\glsentrydesc</code>	135	<code>\glsentrysymbolplural</code>	136
<code>\glsentrydescaccess</code>	315	<code>\glsentrysymbolpluralaccess</code>	315
<code>\Glsentrydescplural</code>	135	<code>\Glsentrytext</code>	135
<code>\glsentrydescplural</code>	135	<code>\glsentrytext</code>	49, 135, 166
<code>\glsentrydescpluralaccess</code>	315	<code>\glsentrytextaccess</code>	314
<code>\Glsentryfirst</code>	136	<code>\glsentrytype</code>	137
<code>\glsentryfirst</code>	136	<code>\Glsentryuseri</code>	137
<code>\glsentryfirstaccess</code>	314	<code>\glsentryuseri</code>	137
<code>\Glsentryfirstplural</code>	136	<code>\Glsentryuserii</code>	137
<code>\glsentryfirstplural</code>	136	<code>\glsentryuserii</code>	137
<code>\glsentryfirstpluralaccess</code>	314	<code>\Glsentryuseriii</code>	137
		<code>\glsentryuseriii</code>	137

<code>\Glsentryuseriv</code>	138	<code>\glslinkvar</code>	93
<code>\glsentryuseriv</code>	138	<code>\glslistdottedwidth</code>	254
<code>\Glsentryuserv</code>	138	<code>\glslocalreset</code>	79
<code>\glsentryuserv</code>	138	<code>\glslocalresetall</code>	80
<code>\Glsentryuservi</code>	138	<code>\glslocalunset</code>	79
<code>\glsentryuservi</code>	138	<code>\glslocalunsetall</code>	80
<code>\glsexpandfields</code>	65	<code>\glslongaccessdisplay</code>	317
<code>\GLSfirst</code>	113	<code>\glslongaccesskey</code>	345
<code>\Glsfirst</code>	113	<code>\glslongkey</code>	196
<code>\glsfirst</code>	113	<code>\glslongpluralaccessdisplay</code>	317
<code>\glsfirstaccessdisplay</code>	316	<code>\glslongpluralaccesskey</code>	345
<code>\GLSfirstplural</code>	115	<code>\glslongpluralkey</code>	196
<code>\Glsfirstplural</code>	115	<code>\glslongtok</code>	200
<code>\glsfirstplural</code>	115	<code>\glsmakefirststuc</code>	244
<code>\glsfirstpluralaccessdisplay</code>	316	<code>\glsmcols</code>	265
<code>\glsgenacfmt</code>	87, 324	<code>\glsmoveentry</code>	76
<code>\glsgenentryfmt</code>	85, 322	<code>\GLSname</code>	116
<code>\glsgetgrouplabel</code>	190	<code>\Glsname</code>	116
<code>\glsgetgrouptitle</code>	143, 189	<code>\glsname</code>	116
<code>\gls glossarymark</code>	9, 37	<code>\glsnameaccessdisplay</code>	316
<code>\gls groupheading</code>	189, 191	<code>\glsnamefont</code>	192, 282
<code>\gls groupskip</code>	189, 191, 251	<code>\glsnavhyperlink</code>	247
<code>\gls hyperlink</code>	141	<code>\glsnavhypertarget</code>	247
<code>\gls hypernavsep</code>	248	<code>\glsnavigation</code>	248
<code>\gls hypernumber</code>	35, 192	<code>\glsnextpages</code>	183
<code>\gls ifhyper</code>	93	<code>\glsnoexpandfields</code>	65
<code>\gls ifhyperon</code>	92	<code>\glsnoidxdisplayloc</code>	179
<code>\gls IfListOfAcronyms</code>	15	<code>\glsnoidxdisplayloclisthandler</code>	179
<code>\gls inlinedescformat</code>	250	<code>\glsnoidxloclist</code>	178
<code>\gls inlinedopostchild</code>	249, 250	<code>\glsnoidxloclisthandler</code>	178
<code>\gls inlineemptydescformat</code>	251	<code>\glsnoidxnumberlistloophandler</code>	157
<code>\gls inlinenameformat</code>	250	<code>\glsnoidxstripaccents</code>	19
<code>\gls inlineparentchildseparator</code>	250	<code>\glsnonextpages</code>	183
<code>\gls inlinepostchild</code>	250	<code>\glsnoxindywarning</code>	39
<code>\gls inlineseparator</code>	250	<code>\glsnumberformat</code>	35
<code>\gls inlinesubdescformat</code>	251	<code>\glsnumberlistloop</code>	157
<code>\gls inlinesubnameformat</code>	251	<code>\glsnumbersgroupname</code>	30, 143, 189
<code>\gls inlinesubseparator</code>	250	<code>\glsnumlistlastsep</code>	141
<code>\gls keylisttok</code>	199	<code>\glsnumlistsep</code>	141
<code>\gls labeltok</code>	200	<code>\glsopenbrace</code>	143
<code>\gls letentryfield</code>	133	<code>\glsorder</code>	23
<code>\gls link</code>	81, 93, 106, 141, 190, 192	<code>\glsorg@endtheglossary</code>	5
<code>\gls link options</code>		<code>\glsorg@theglossary</code>	5
counter	92, 106, 236	<code>\gls pagelistwidth</code>	254, 260, 269, 276
format	92, 106, 192	<code>\glspar</code>	33
hyper	92, 94, 106	<code>\glspercentchar</code>	143
local	92	<code>\GLSpl</code>	110
<code>\gls linkcheckfirsthyperhook</code>	94		

<code>\Glspl</code>	109, 243	<code>\glssubentrycounterlabel</code> ...	185
<code>\glspl</code>	81, 109, 110	<code>\glssubentryitem</code>	185
<code>\GLSplural</code>	114	<code>\GLSsymbol</code>	119
<code>\Glsplural</code>	114	<code>\Glsymbol</code>	119
<code>\glsplural</code>	114	<code>\glsymbol</code>	118, 119
<code>\glspluralaccessdisplay</code>	316	<code>\glsymbolaccessdisplay</code>	316
<code>\glspluralsuffix</code>	30, 59	<code>\glsymbolnav</code>	248
<code>\glspostdescription</code>	9	<code>\GLSsymbolplural</code>	120
<code>\glspostinline</code>	250	<code>\Glsymbolplural</code>	119
<code>\glsprestandardsort</code>	10	<code>\glsymbolplural</code>	119, 120
<code>\glsquote</code>	143	<code>\glsymbolpluralaccessdisplay</code>	
<code>\glsrefentry</code>	184	317
<code>\glsreset</code>	79	<code>\glsymbolsgroupname</code> .	30, 143, 189
<code>\glsresetall</code>	80	<code>\glstarget</code>	186
<code>\glsresetentrylist</code>	183	<code>\GLStext</code>	112
<code>\glsresetsubentrycounter</code> ...	184	<code>\Glstext</code>	113
<code>\glssee</code>	165	<code>\glstext</code>	112
<code>\glsseeformat</code>	145, 165	<code>\glstextaccessdisplay</code>	316
<code>\glsseeitem</code>	166	<code>\glstextformat</code>	81
<code>\glsseeitemformat</code>	166	<code>\glstextup</code>	196
<code>\glsseeelastsep</code>	165	<code>\glstildechar</code>	143
<code>\glsseeelist</code>	165	<code>\glstreeindent</code>	284–286
<code>\glsseesep</code>	166	<code>\glstreenamefmt</code>	282
<code>\glsSetAlphaCompositor</code>	34	<code>\glsunset</code>	79
<code>\glsSetCompositor</code>	34	<code>\glsunsetall</code>	80
<code>\glssetexpandfield</code>	17	<code>\GlsUseAcrEntryDispStyle</code> ...	202
<code>\glssetnoexpandfield</code>	17	<code>\GlsUseAcrStyleDefs</code>	202
<code>\glsSetSuffixF</code>	34	<code>\GLSuseri</code>	121
<code>\glsSetSuffixFF</code>	35	<code>\Glsuseri</code>	120
<code>\glssettoctitle</code>	30	<code>\glsuseri</code>	120, 121
<code>\glssetwidest</code>	287	<code>\GLSuserii</code>	122
<code>\GlsSetXdyCodePage</code>	47	<code>\Glsuserii</code>	121
<code>\GlsSetXdyFirstLetterAfterDigits</code>		<code>\glsuserii</code>	121, 122
.....	143	<code>\GLSuseriii</code>	122
<code>\GlsSetXdyLanguage</code>	47	<code>\Glsuseriii</code>	122
<code>\GlsSetXdyLocationClassOrder</code>	46	<code>\glsuseriii</code>	122
<code>\GlsSetXdyMinRangeLength</code> ...	144	<code>\GLSuseriv</code>	123
<code>\GlsSetXdyStyles</code>	46	<code>\Glsuseriv</code>	123
<code>\glsshortaccessdisplay</code>	317	<code>\glsuseriv</code>	123
<code>\glsshortaccesskey</code>	345	<code>\GLSuserv</code>	124
<code>\glsshortkey</code>	196	<code>\Glsuserv</code>	124
<code>\glsshortpluralaccessdisplay</code>	317	<code>\glsuserv</code>	124
<code>\glsshortpluralaccesskey</code> ...	345	<code>\GLSuservi</code>	125
<code>\glsshortpluralkey</code>	196	<code>\Glsuservi</code>	125
<code>\glsshorttok</code>	200	<code>\glsuservi</code>	125
<code>\glssortnumberfmt</code>	11	<code>\glswrite</code>	153
<code>\glsspace</code>	197	<code>\glswritedefhook</code>	70
<code>\glsstepentry</code>	184	<code>\gmFUNocap</code>	245
<code>\glsstepsubentry</code>	184		

H

<code>\hyperbf</code>	194
<code>\hyperemph</code>	195
<code>hyperfirst</code> (option)	22
<code>\hyperit</code>	194
<code>\hyperlink</code>	105
<code>\hypermd</code>	194
<code>\hyperpage</code>	192
<code>hyperref</code> package ...	163, 166, 192, 236
<code>\hyperrm</code>	194
<code>\hypersc</code>	195
<code>\hypersf</code>	194
<code>\hypersl</code>	194
<code>\hypertarget</code>	105
<code>\hypertt</code>	194
<code>\hyperup</code>	195

I

<code>\if@gl@docloaded</code>	5
<code>\if@gl@isacronymlist</code>	15
<code>\ifglossaryexists</code>	49
<code>\ifgl@descsuppressed</code>	52
<code>\ifgl@sentryexists</code>	50
<code>\ifgl@shaschildren</code>	51
<code>\ifgl@shasdesc</code>	52
<code>\ifgl@shasfield</code>	53
<code>\ifgl@shaslong</code>	52
<code>\ifgl@shasparent</code>	51
<code>\ifgl@shasprefix</code>	239
<code>\ifgl@shasprefixfirst</code>	239
<code>\ifgl@shasprefixfirstplural</code> ..	239
<code>\ifgl@shasprefixplural</code>	239
<code>\ifgl@shasshort</code>	52
<code>\ifgl@shassymbol</code>	52
<code>\ifgl@stranslate</code>	22
<code>\ifgl@sused</code>	50, 78
<code>\ifgl@xindy</code>	24
<code>\ifignoredglossary</code>	58
<code>index</code> (option)	27
<code>index</code> (style)	282
<code>indexgroup</code> (style)	283
<code>indexhypergroup</code> (style)	283
<code>indexonlyfirst</code> (option)	22
<code>inline</code> (style)	249
<code>\inputencodingname</code>	24
<code>\istfile</code>	158
<code>\istfilename</code>	33
<code>\item</code>	192, 251, 282, 283

L

<code>link</code> text	81
<code>list</code> (style)	251
<code>listdotted</code> (style)	253
<code>listgroup</code> (style)	252
<code>listhypergroup</code> (style)	252
<code>\loadglsentries</code>	13, 81
<code>long</code> (key)	62
<code>long</code> (style)	254
<code>long-sc-short</code> (acrstyle)	204
<code>long-sc-short-desc</code> (acrstyle) ..	205, 332
<code>long-short</code> (acrstyle)	203, 331
<code>long-short-desc</code> (acrstyle) ..	205, 332
<code>long-sm-short</code> (acrstyle)	204
<code>long-sm-short-desc</code> (acrstyle) ..	205, 333
<code>long3col</code> (style)	256
<code>long3colborder</code> (style)	256
<code>long3colheader</code> (style)	257
<code>long3colheaderborder</code> (style) ..	257
<code>long4col</code> (style)	257
<code>long4colborder</code> (style)	258
<code>long4colheader</code> (style)	258
<code>long4colheaderborder</code> (style) ..	258
<code>longaccess</code> (key)	312
<code>longborder</code> (style)	255
<code>longheader</code> (style)	255
<code>longheaderborder</code> (style)	255
<code>\longnewglossaryentry</code> ...	58, 70
<code>longplural</code> (key)	62
<code>longpluralaccess</code> (key)	312
<code>\longprovideglossaryentry</code> ...	70
<code>longragged</code> (style)	260
<code>longragged3col</code> (style)	262
<code>longragged3colborder</code> (style) ..	263
<code>longragged3colheader</code> (style) ..	263
<code>longragged3colheaderborder</code> (style)	263
<code>longraggedborder</code> (style)	261
<code>longraggedheader</code> (style)	261
<code>longraggedheaderborder</code> (style) ..	262
<code>longtable</code> (environment)	8, 230, 254–265
<code>longtable</code> package	254, 260

M

<code>\makefirstuc</code>	243
<code>makeglossaries</code>	23, 24, 33, 47, 55, 152, 169

<code>\makeglossaries</code>	
... 25, 29, 33, 34, 54, 56, 152, 153	
<code>\makeglossary</code>	153
<code>makeindex</code>	359
<code>makeindex</code>	
. 10, 24, 25, 30, 33–35, 55, 56,	
59, 78, 97, 100, 142, 146, 148–	
150, 159, 163, 188, 189, 291, 292	
<code>delim_n</code>	35
<code>delim_r</code>	35
<code>page_compositor</code>	34
special characters	98, 99, 142
<code>makeindex (option)</code>	24
<code>\makenoidxglossaries</code>	21, 154
<code>mcolalttree (style)</code>	268
<code>mcolalttreegroup (style)</code>	268
<code>mcolalttreehypergroup (style)</code> .	269
<code>mcolindex (style)</code>	266
<code>mcolindexgroup (style)</code>	266
<code>mcolindexhypergroup (style)</code> ...	266
<code>mcoltree (style)</code>	266
<code>mcoltreegroup (style)</code>	267
<code>mcoltreehypergroup (style)</code>	267
<code>mcoltreename (style)</code>	267
<code>mcoltreenamegroup (style)</code> ...	267
<code>mcoltreenamehypergroup</code>	
(style)	268
<code>memoir class</code>	159
<code>mfirstuc package</code>	1, 246
<code>\mfirstucMakeUppercase</code>	244
<code>\mfu@checkword</code>	245
<code>\MFUclear</code>	246
<code>\MFUnocap</code>	245
<code>multicol package</code>	265
<code>multicols (environment)</code>	266
N	
<code>name (key)</code>	58
<code>\new@glossaryentry</code>	65
<code>\newacronym</code> .. 23, 62, 80, 81, 195, 196	
<code>\newacronymhook</code>	200
<code>\newacronymstyle</code>	202
<code>\newglossary</code> 16, 55, 56, 150, 152, 180	
<code>\newglossaryentry</code> 58, 65, 80, 81, 196	
<code>\newglossaryentry options</code>	
<code>access</code>	313, 314
<code>counter</code>	60
<code>description</code>	23, 58, 62,
65, 71, 116, 134, 135, 196, 223, 311	
<code>descriptionaccess</code>	315, 317
<code>descriptionplural</code>	117, 312
<code>descriptionpluralaccess</code> ..	315, 317
<code>first</code>	59,
74, 106, 113, 136, 221, 226, 311	
<code>firstaccess</code>	314, 316
<code>firstplural</code>	59, 115, 136, 311
<code>firstpluralaccess</code>	314, 316
<code>format</code>	144
<code>long</code>	87, 139, 312
<code>longaccess</code>	315, 317
<code>longplural</code>	139, 312
<code>longpluralaccess</code>	315, 317
<code>name</code>	58,
59, 62, 65, 71, 116, 133, 166, 311	
<code>nonumberlist</code>	61
<code>parent</code>	61, 65
<code>plural</code>	59, 74, 114, 311
<code>pluralaccess</code>	314, 316
<code>prefix</code>	237
<code>prefixfirst</code>	237
<code>prefixfirstplural</code>	238
<code>prefixplural</code>	238
<code>see</code>	8, 60, 152, 154
<code>short</code>	87, 138, 312
<code>shortaccess</code>	315, 317
<code>shortplural</code>	138, 312
<code>shortpluralaccess</code>	315, 317
<code>sort</code>	59, 137, 188, 189
<code>symbol</code>	58, 59, 118, 216–
218, 221, 226, 257, 273, 311, 313	
<code>symbolaccess</code>	314, 316
<code>symbolplural</code>	119, 311
<code>symbolpluralaccess</code>	315, 317
<code>text</code> 59, 106, 112, 135, 216, 221, 311	
<code>textaccess</code>	314, 316
<code>type</code>	13, 60, 81, 137
<code>user1</code>	120, 137, 312
<code>user2</code>	121, 137
<code>user3</code>	122, 137
<code>user4</code>	123, 138
<code>user5</code>	124, 138
<code>user6</code>	125, 138, 312
<code>\newglossarystyle</code>	191
<code>\newignoredglossary</code>	57
<code>nogroupskip (option)</code>	9
<code>nohypertypes (option)</code>	16
<code>\noist</code>	149, 236, 296
<code>nolist (option)</code>	8

nolong (option)	8
nomain (option)	13
nonumberlist (key)	61
nonumberlist (option)	7
\nopostdesc	32
nopostdot (option)	9
noredefwarn (option)	17
nostyles (option)	9
nosuper (option)	8
notranslate (option)	22
notree (option)	9
nowarn (option)	17
\ns@newglossary	55
numberedsection (option)	6
numberline (option)	6
numbers (option)	26

O

\oldacronym	195
order (option)	24

P

\p@glshyp@opt	93
package options:	
acronym	13, 14, 30, 167, 196
true	14
acronym	13
acronymlists	16
acronyms	14
automake	25
compatible-2.07	26
compatible-3.07	26
counter	16
counter	16
description	221
description	23
dua	219, 221
dua	23
entrycounter	181, 183
true	10
entrycounter	9
entrycounterwithin	10
footnote	107–111, 217, 219, 221, 223
footnote	23
hyperfirst	
false	107–111
hyperfirst	22
index	27
indexonlyfirst	367
indexonlyfirst	22

makeindex	146, 236
makeindex	24
nogroupskip	9
nohypertypes	16
nolist	230
nolist	8
nolong	230, 254
nolong	8
nomain	12, 13
nomain	13
nonumberlist	7
nonumberlist	7
nopostdot	9
noredefwarn	17
nostyles	9
nosuper	231
nosuper	8
notranslate	22
notree	231
notree	9
nowarn	17
numberedsection	6
numberline	6
numberline	6
numbers	26
order	24
sanitize	20, 58, 133, 135
sanitize	21
sanitizesort	20
savenuumberlist	8
savewrites	25, 364, 365
false	150
true	152, 158
savewrites	25
section	6, 37
section	6
seeautonumberlist	8
shotcuts	23
smallcaps	23
smaller	23
sort	
def	10, 11
standard	10
use	10, 11
sort	10
style	7, 230, 231
style	7
subentrycounter	181, 183
subentrycounter	10

symbols	26
toc	6
true	6
toc	6
translate	22
false	22
translate	22
translator	22
ucmark	9
xindy	24, 25, 146, 236
xindy	24
xindygloss	25
xindynoglsnumbers	25
\pagelistname	30
parent (key)	61
\PGLS	242
\Pgls	240
\pgls	239
\PGLSpl	242
\Pglspl	241
\pglspl	240
\phantomsection	36–38
plural (key)	59
pluralaccess (key)	311
polyglossia package	28, 31
\printacronyms	14
\PrintChanges	5
\printglossaries	
. 12, 35, 54, 57, 153, 166, 167, 246	
\printglossary	35, 36,
55, 57, 153, 166, 169, 179, 246, 247	
\printglossary options	
entrycounter	181
nogroupskip	180
nonumberlist	182
nopostdot	181
numberedsection	180
style	180
subentrycounter	181
title	180
toctitle	180
type	13, 166, 179
\printnoidxglossaries	167
\printnoidxglossary	167, 179
\printnoidxglossary options	
sort	182
\provideglossaryentry	65
R	
\renewacronymstyle	202
\renewglossarystyle	192
\roman	43
S	
\s@glshyp@opt	93
\s@newglossary	55
sanitize (option)	21
sanitizesort (option)	20
savenumberlist (option)	8
savewrites (option)	25
sc-short-long (acrstyle)	204
sc-short-long-desc (acrstyle) ..	
.....	206, 333
\scantokens	49
\section	49
section (option)	6
see (key)	60
seeautonumberlist (option)	8
\seename	31
\SetAcronymLists	15
\SetAcronymStyle	14, 15, 228
\setacronymstyle	201
\SetCustomDisplayStyle	229
\SetCustomStyle	230
\SetDefaultAcronymDisplayStyle	
.....	213
\SetDefaultAcronymStyle	214
\SetDescriptionAcronymDisplayStyle	
.....	219
\SetDescriptionAcronymStyle	
.....	221
\SetDescriptionDUAAcronymDisplayStyle	
.....	217
\SetDescriptionDUAAcronymStyle	
.....	218
\SetDescriptionFootnoteAcronymDisplayStyle	
.....	215
\SetDescriptionFootnoteAcronymStyle	
.....	216
\SetDUADisplayStyle	227
\SetDUASStyle	228
\setentrycounter	190
\SetFootnoteAcronymDisplayStyle	
.....	221
\SetFootnoteAcronymStyle ...	223
\SetGenericNewAcronym	200
\setglossarypreamble	36
\setglossarysection	37
\setglossarystyle	191
\setglossentrycompatibility	
.....	188

<code>\SetSmallAcronymDisplayStyle</code>	224
<code>\SetSmallAcronymStyle</code>	226
<code>\setStyleFile</code>	33
<code>\setupglossaries</code>	27
<code>short (key)</code>	62
<code>short-long (acrstyle)</code>	203, 331
<code>short-long-desc (acrstyle)</code>	206, 333
<code>shortaccess (key)</code>	312
<code>shortplural (key)</code>	62
<code>shortpluralaccess (key)</code>	312
<code>shotcuts (option)</code>	23
<code>\showacronymlists</code>	235
<code>\showglocounter</code>	232
<code>\showgloDESC</code>	233
<code>\showgloDESCaccess</code>	346
<code>\showgloDESCplural</code>	234
<code>\showgloDESCpluralaccess</code>	346
<code>\showgloFIRST</code>	232
<code>\showgloFIRSTaccess</code>	345
<code>\showgloFIRSTpl</code>	232
<code>\showgloFIRSTpluralaccess</code>	345
<code>\showgloFLAG</code>	235
<code>\showgloINDEX</code>	235
<code>\showgloLEVEL</code>	231
<code>\showgloLOCLIST</code>	235
<code>\showgloLONG</code>	234
<code>\showgloLONGaccess</code>	346
<code>\showgloLONGpluralaccess</code>	346
<code>\showgloNAME</code>	233
<code>\showgloNAMEaccess</code>	345
<code>\showgloPARENT</code>	231
<code>\showgloPLURAL</code>	231
<code>\showgloPLURALaccess</code>	345
<code>\showgloSHORT</code>	234
<code>\showgloSHORTaccess</code>	346
<code>\showgloSHORTpluralaccess</code>	346
<code>\showgloSORT</code>	234
<code>\showglossaries</code>	235
<code>\showglossarycounter</code>	236
<code>\showglossaryentries</code>	236
<code>\showglossaryin</code>	235
<code>\showglossaryout</code>	236
<code>\showglossarytitle</code>	236
<code>\showgloSYMBOL</code>	234
<code>\showgloSYMBOLaccess</code>	346
<code>\showgloSYMBOLplural</code>	234
<code>\showgloSYMBOLpluralaccess</code>	346
<code>\showgloTEXT</code>	231
<code>\showgloTEXTaccess</code>	345
<code>\showgloTYPE</code>	232
<code>\showgloUSERI</code>	232
<code>\showgloUSERII</code>	232
<code>\showgloUSERIII</code>	233
<code>\showgloUSERIV</code>	233
<code>\showgloUSERV</code>	233
<code>\showgloUSERVI</code>	233
<code>sm-short-long (acrstyle)</code>	204
<code>sm-short-long-desc (acrstyle)</code>	206, 334
<code>smallcaps (option)</code>	23
<code>smaller (option)</code>	23
<code>\SmallNewAcronymDef</code>	225, 344
<code>sort (key)</code>	59
<code>sort (option)</code>	10
<code>style (option)</code>	7
<code>subentrycounter (option)</code>	10
<code>\subglossentry</code>	186
<code>\subitem</code>	283
<code>sublistdotted (style)</code>	254
<code>\subsubitem</code>	283
<code>super (style)</code>	269
<code>super3col (style)</code>	271
<code>super3colborder (style)</code>	272
<code>super3colheader (style)</code>	272
<code>super3colheaderborder (style)</code>	272
<code>super4col (style)</code>	273
<code>super4colborder (style)</code>	274
<code>super4colheader (style)</code>	274
<code>super4colheaderborder (style)</code>	274
<code>superborder (style)</code>	270
<code>superheader (style)</code>	270
<code>superheaderborder (style)</code>	271
<code>superragged (style)</code>	276
<code>superragged3col (style)</code>	278
<code>superragged3colborder (style)</code>	279
<code>superragged3colheader (style)</code>	279
<code>superraggedborder (style)</code>	277
<code>superraggedheader (style)</code>	277
<code>superraggedheaderborder (style)</code>	278
<code>superraggedright3colheaderborder (style)</code>	280
<code>supertabular (environment)</code>	8, 231, 269–282
<code>supertabular package</code>	8, 231, 269, 276
<code>symbol (key)</code>	59
<code>symbolaccess (key)</code>	311
<code>\symbolname</code>	30
<code>symbolplural (key)</code>	60

symbolpluralaccess (key) 311
symbols (option) 26

T

text (key) 59
textaccess (key) 311
textcase package 4
\theequation 163
theglossary (environment)
..... 5, 17, 35, 36,
185, 186, 191, 267, 268, 284–287
\theHequation 163
theindex (environment) 282
toc (option) 6
\translate 31
translate (option) 22
translator package 13,
14, 26–28, 31, 166, 347, 356–359
tree (style) 284
treegroup (style) 285
treehypergroup (style) 285
treenoname (style) 285
treenonamegroup (style) 286
treenonamehypergroup (style) .. 286
type (key) 60

U

ucmark (option) 9
user1 (key) 61
user2 (key) 61
user3 (key) 61
user4 (key) 61
user5 (key) 61
user6 (key) 61

W

\warn@nomakeglossaries 151
\warn@noprintglossary 166
\writeist 33, 40, 42, 45, 144, 290, 292

X

\xcapitalisewords 246
\xglsacccsupp 316
xindy 359
xindy 10, 24, 25, 33, 34, 39, 43, 45–
48, 77, 103, 104, 143–145, 159,
163, 169, 188, 189, 236, 291, 292
xindy (option) 24
xindygloss (option) 25
xindynoglsnumbers (option) 25
\xmakefirstuc 245
xspace package 5, 195